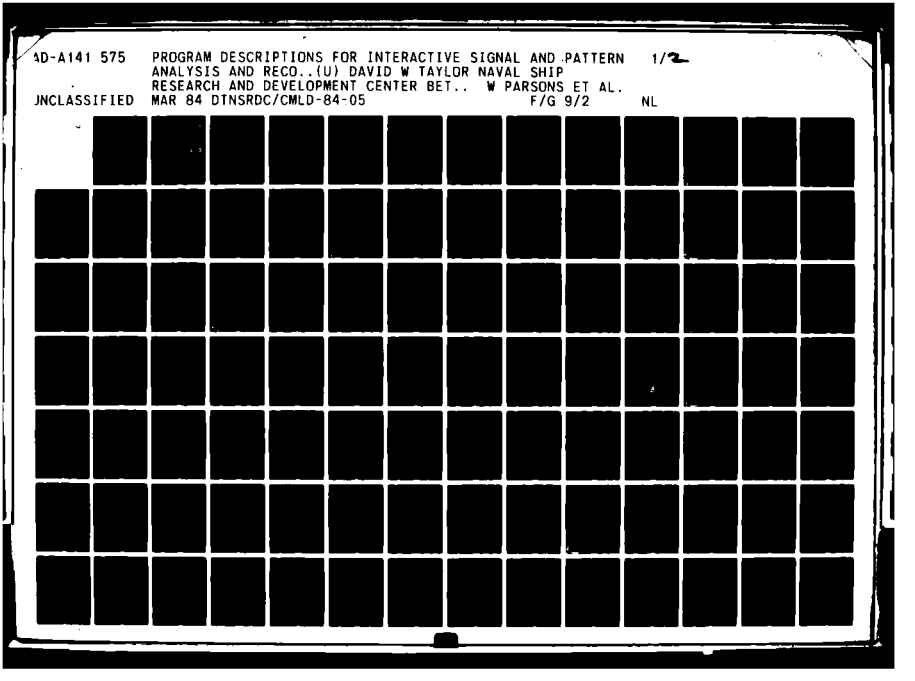
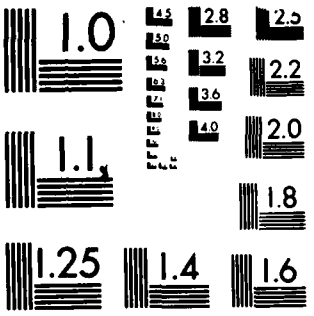
AD-A141 575 PROGRAM DESCRIPTIONS FOR INTERACTIVE SIGNAL AND PATTERN 1/2
ANALYSIS AND RECO..(U) DAVID W TAYLOR NAVAL SHIP
RESEARCH AND DEVELOPMENT CENTER BET.. W PARSONS ET AL.
UNCLASSIFIED MAR 84 DTNSRDC/CMLD-84-05 F/G 9/2 NL





(12)

DTNSRDC/CMLD-84/05

DAVID W. TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CENTER

Bethesda, Maryland 20084

PROGRAM DESCRIPTIONS FOR INTERACTIVE SIGNAL AND PATTERN ANALYSIS AND RECOGNITION SYSTEM (ISPARS)

by

William Parsons
Joseph Garner
James Carlberg
Sidney Berkowitz

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

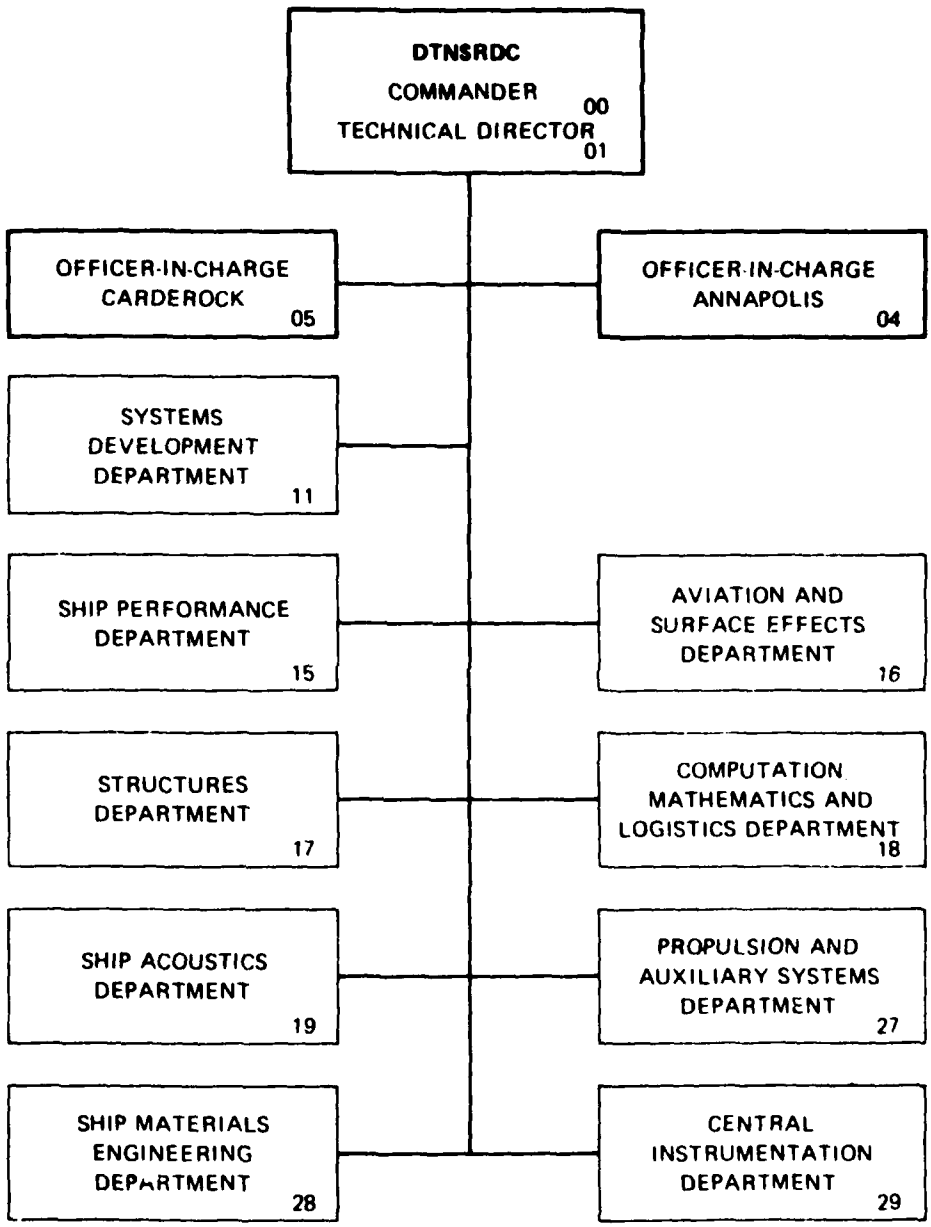
COMPUTATION, MATHEMATICS, AND LOGISTICS DEPARTMENT
DEPARTMENTAL REPORT

March 1984

DTNSRDC/CMLD-84/05

PROGRAM DESCRIPTIONS FOR INTERACTIVE SIGNAL AND PATTERN ANALYSIS AND RECOGNITION SYSTEM (ISPARS)

MAJOR DTNSRDC ORGANIZATIONAL COMPONENTS



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: DTNSRDC/CMLD-84/05	2. GOVT ACCESSION NO.: AD-A142 575
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE *(and Subtitle)*: PROGRAM DESCRIPTIONS FOR INTERACTIVE SIGNAL AND PATTERN ANALYSIS AND RECOGNITION SYSTEM (ISPARS)	5. TYPE OF REPORT & PERIOD COVERED: Interim Report Sep 1979 - Sep 1980
	6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): William Parsons, James Carlberg, Joseph Garner, Sidney Berkowitz	8. CONTRACT OR GRANT NUMBER(s):
9. PERFORMING ORGANIZATION NAME AND ADDRESS: David Taylor Naval Ship Research and Development Center, Bethesda, Maryland 20084	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: (See Reverse Side)
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Sea Systems Command, Materials and Mechanics Division, Washington, DC 20362	12. REPORT DATE: March 1984
	13. NUMBER OF PAGES: 138
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*:	15. SECURITY CLASS. *(of this report)*: UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT *(of this Report)*

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Signal Analysis
Pattern Recognition
Spectral Analysis
Interactive Graphics
Waveform Processing

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The Interactive Signal and Pattern Analysis and Recognition System (ISPARS) is an integrated package of interactive graphics software and signal and pattern analysis programs for visualizing and exploring sampled data streams and designing pattern logics. This report contains detailed descriptions of the subroutines, files, and linking procedures for the ISPARS components developed at the David Taylor Naval Ship Research and Development Center (DTNSRDC) which are not documented in other DTNSRDC reports.

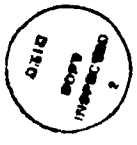
DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

(Block 10)

Program Element 61153N
Project SR01403
Task Area SR0140301
Work Unit 1808-010



Accession For	
NTIS GRA&I	☑
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

The Interactive Signal and Pattern Analysis and Recognition System (ISPARS) is an integrated package of interactive graphics software and signal and pattern analysis programs for visualizing and exploring sampled data streams and designing pattern logics. This report contains detailed descriptions of the subroutines, files, and linking procedures for the ISPARS components developed at the David Taylor Naval Ship Research and Development Center (DTNSRDC) which are not documented in other DTNSRDC reports.

ADMINISTRATIVE INFORMATION

This work was completed in the Computer Science and Information Systems Division of the Computation, Mathematics, and Logistics Department under the sponsorship of NAVSEA 03F, Task Area SR 0140301, Task 15321, Element 61153N.

INTRODUCTION

The Interactive Signal and Pattern Analysis and Recognition System (ISPARS) is an integrated package of interactive graphics software and signal and pattern analysis programs for visualizing and exploring sampled data streams and designing pattern logics. An ISPARS User's Guide[1*] presents the conceptual descriptions and functional guide to the system. This report is a companion to the User's Guide and contains detailed descriptions of the subroutines, files, and linking procedures for the ISPARS components developed at the David Taylor Naval Ship Reserach and Development Center (DTNSRDC), which are not documented in other DTNSRDC reports.

ISPARS deals with two areas of data analysis: The processing of sampled signal waveforms and the determination of pattern classes into which waveforms may be separated. Signal analysis encompasses segmentation, spectral decomposition, filtering, normalization, and characterization of signals. It provides insight into the physical sources of signal components and the identification and extraction of significant features in the signal. The pattern analysis procedures enhance the understanding of pattern relationships and permits the construction of classification criteria for recognition schemes. ISPARS integrates these analysis techniques into a set of interactive graphics

---

* A complete list of references is given on page 133.

programs which provide convenient, flexible means of visualizing, exploring, and analyzing a sampled data stream controlled through a simple, concise command language.

This report is divided into three main sections corresponding to three subsystems of ISPARS: VU, SELECT, and WAVAN. Each contains program descriptions, file descriptions, and the linking procedure used in program development. The remaining subsystems and components of ISPARS are documented in other reports.[2,3,4]

PROGRAM VU

VU is an interactive, general purpose program for the examination and analysis of digitized signals. VU runs on a PDP 11/45 minicomputer system with a VT11 processor supported by the RT-11 operating system. Digitized data are stored on disk or magnetic tape as an infinite data stream, and a graphics display monitor is used to show results of analyses, spectral decomposition, peak-picking, and smoothing. A functional description of VU and its command language is contained in the ISPARS User's Guide[1]. This section presents, in alphabetical order by subroutine name, the subroutine descriptions, input/output, files, and linking procedures associated with program VU.

VU SUBROUTINE DESCRIPTIONS

SUBROUTINE NAME ADJUST
PURPOSE To display and alter parameter values for VU.
DESCRIPTION
INPUT
OUTPUT
CALLED BY VU GRAPHICS? Yes
SUBROUTINES CALLED AJ1SUB DISK I/O? Yes
CALLING SEQUENCE CALL ADJUST [DPO:OPARAM.DAT]

SUBROUTINE NAME AJ1SUB
PURPOSE To display parameters for ADJUST.
DESCRIPTION
INPUT
OUTPUT
CALLED BY *ADJUST* GRAPHICS? Yes
SUBROUTINES CALLED None DISK I/O? No
CALLING SEQUENCE CALL AJ1SUB (AMPMIN, SFREQ, LOGPOW, AMPBEG, BAND, PERHI, PERLO)

SUBROUTINE NAME BESSEL
PURPOSE To calculate a modified Bessel function.
DESCRIPTION
INPUT
OUTPUT
CALLED BY POCALC GRAPHICS? No
SUBROUTINES CALLED None DISK I/O? No
CALLING SEQUENCE CALL BESSEL (XUA-UUE, ZSUM)

SUBROUTINE NAME CHANGE

PURPOSE To change KLASIT parameters.

DESCRIPTION CHANGE is a FORTRAN subroutine which receives a change parameter command and new value as arguments and changes the specified parameter value contained in common block/PARAMS/. CHANGE also displays the new parameter value and erases all displays associated with the parameter before returning to the calling program.

INPUT

Arguments

CMAND 3 - first letter of command. Valid letters are "N", "L", "I", "P", and "Q".

NUM3 - second character of command; will be a number 1, 2, or 3 referring to parameter set 1, 2, or 3.

ZNUM3 - new floating point value of specified parameter

OUTPUT New parameter value displayed along with other parameters in its parameter set.

CALLED BY KLOOK

SUBROUTINES CALLED None

CALLING SEQUENCE CALL CHANGE (CMAND3, NUM3, ZNUM3)

SUBROUTINE NAME CHINFO

PURPOSE To obtain channel information from the user and supply it to VU.

DESCRIPTION

INPUT

OUTPUT

CALLED BY VU GRAPHICS? No

SUBROUTINES CALLED SHOUT DISK I/O? No

CALLING SEQUENCE CALL CHINFO (ICHAN, OCHAN, NCHAN, ZLASL, FILSIZ, CMAND1)

SUBROUTINE NAME CREAT1

PURPOSE To calculate the unsmoothed noise from the results of applying KLASIT to the original data.

DESCRIPTION Subroutine CREAT1 obtains an estimate of the noise contained in the original data by examining the results of the first application of subroutine KLASIT to the original data. The rise, level, and fall forms generated by KLASIT are contained on disk file DPO:UNSMOO.DAT. The output array RUFF is filled in with a constant noise level associated with each form, with the index to RUFF beginning at the start time for the form and running up to (but not including) the start time for the next form. A zero noise is associated with all level forms and any rise or fall with durations exceeding Q1 points where Q1 and Q0 are input parameters. For short rises or falls of duration less than Q0, the noise is taken to be the entire absolute change in level. For durations between Q0 and Q1 the noise is a percentage ((Q1-duration) divided by (Q1-Q0)) of the change in level. The flowchart for CREAT1 is shown in Figure 1.

INPUT

Arguments

Q0, Q1 - duration parameters Q0<Q1

Disk

DPO:UNSMOO.DAT - file containing KLASIT approximation to original data

OUTPUT

Arguments

RUFF - 1024-point array containing the unsmoothed noise estimate

CALLED BY KLOOK

SUBROUTINES CALLED None

CALLING SEQUENCE CALL CREAT1 (RUFF, Q0, Q1)

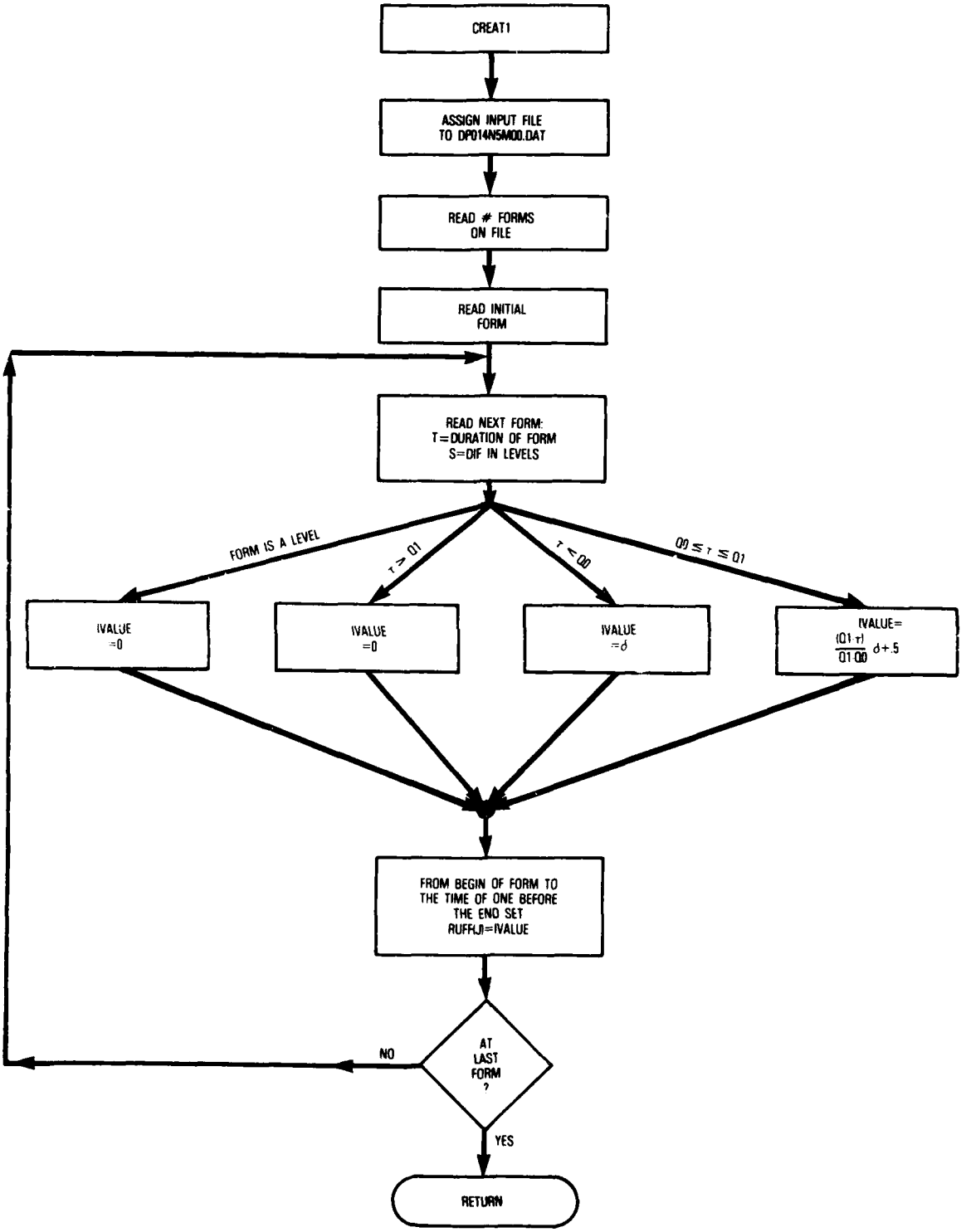


Figure 1 - Subroutine CREAT1 Flowchart

SUBROUTINE NAME CREAT2

PURPOSE To construct the noise array from the smoothed estimate of the noise.

DESCRIPTION Subroutine CREAT2 examines the smoothed estimate of the noise produced by KLASIT on file DPO:SMOO.DAT and computes a final estimate of the noise array. For all points contained in a level form, the noise is set to one-half the magnitude of the level. For points in a rise or fall the noise is obtained from a straight line interpolation between the levels divided by two. The division by two is to account for positive and negative values. The flowchart for CREAT2 is shown in Figure 2.

INPUT

Disk

DPO:SMOO.DAT - file containing KLASIT approximation to unsmoothed noise estimate

OUTPUT

Arguments

NOISE - A 1024-word array contining final smoothed estimate of noise contained in original data

MXSMOO - twice maximum value of noise array

Printout: Noise array and MXSMOO

CALLED BY KLOOK

SUBROUTINES CALLED None

CALLING SEQUENCE CALL CREAT2 (NOISE, MXSMOO)

SUBROUTINE NAME DISKRD

PURPOSE To fill ARRAY with data from a specified disk file.

DESCRIPTION

INPUT

OUTPUT

CALLED BY (KLOOK, VU, SEARCH) GRAPHICS? No

SUBROUTINES CALLED REEDTR DISK I/O? Yes

[DPO:TRAKS2.DAT]

CALLING SEQUENCE CALL DISKRD (ARRAY, TRKNUM, CMAND4, NUM4, ZNUM4)

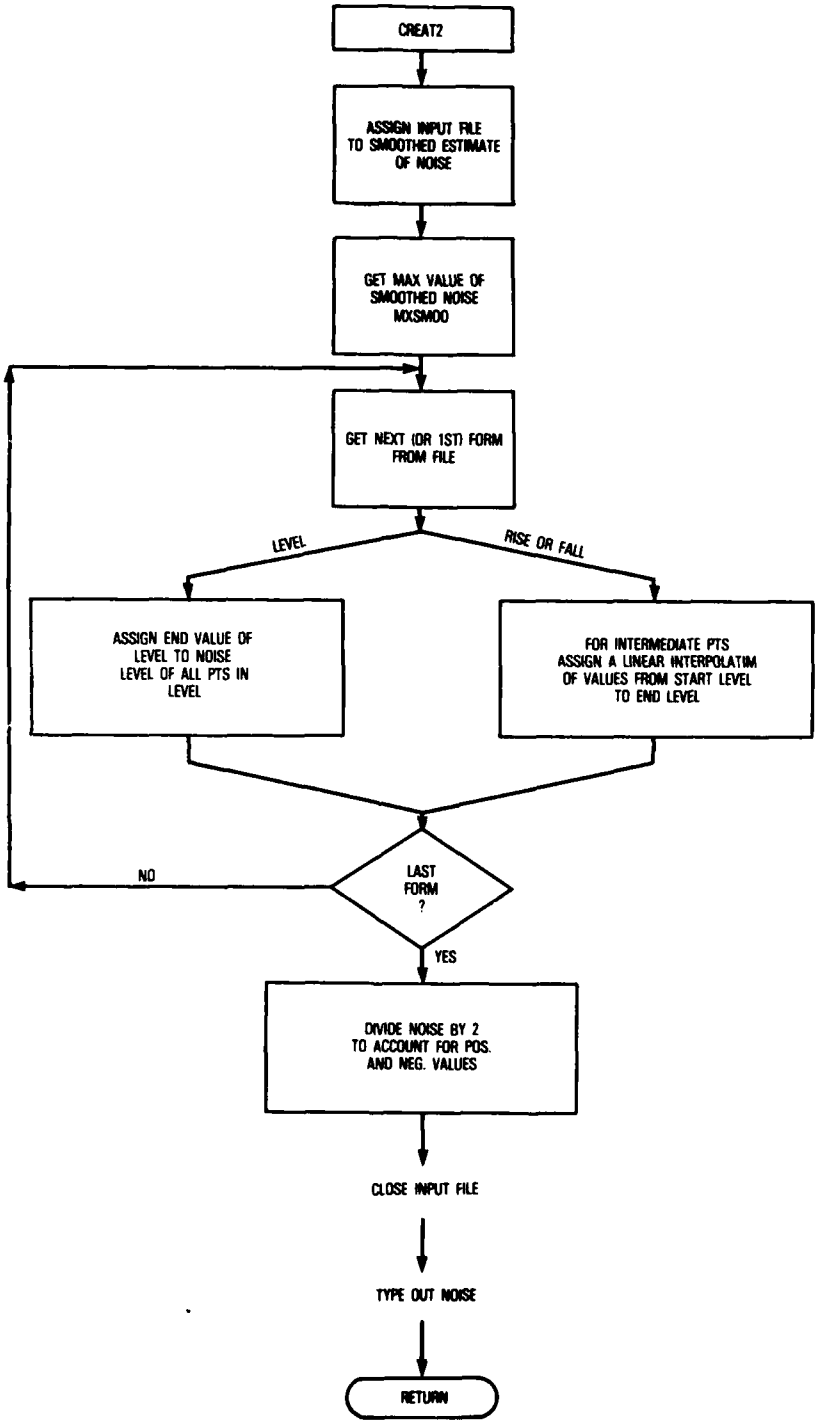


Figure 2 - Subroutine CREAT2 Flowchart

SUBROUTINE NAME DSPIT1

PURPOSE To display the unsmoothed noise array.

DESCRIPTION DSPIT1 calls subroutine NEWVEC in creating a subpicture display of the unsmoothed noise array which it receives as an input argument. The subpicture is turned on and a copy of the subpicture is made to be displayed at a higher point on the screen should the "merge" command be given to KLOOK. Finally, DSPIT1 will turn off the unsmoothed noise display if the last command given is a D3 or any non "D" command. The final status of the unsmoothed noise display and its copy are returned in variables BRUFF and TRUFF of common /DCHEK/.

INPUT

Arguments

RUFF - a maximum 1024-point array containing the unsmoothed noise data

LNTH - length of RUFF

Common

/CMNDS/

CMAND3 - 1st character of last user command

NUM3 - 2nd character of last user command

/SCALE/

ZR - small factor

OUTPUT

Common

/DCHEK/ BRUFF, TRUFF - status variables of unsmoothed noise display and its merge copy

CALLED BY KLOOK

SUBROUTINES CALLED NEWVEC

CALLING SEQUENCE CALL DSPIT1 (RUFF, LNTH)

SUBROUTINE DSPIT2

PURPOSE To display the smoothed noise.

DESCRIPTION DSPIT2 reads from disk the results of subroutine KLASIT operating on the unsmoothed noise (sequence of rise, level, and fall waveforms referred to as the smoothed noise). By computing y-differences and cycling through calls to subroutine NEWVEC, DSPIT2 generates a subpicture display of the unsmoothed noise which is placed in the top part of the screen and assigned the status variable and tag TSMOO. Before returning, the smoothed noise display is turned off unless the last user command was "D2".

INPUT

Disk

DPO:SMOO.DAT - FORTRAN direct access file containing up to 1024 records of three words each where each record represents a waveform approximation by KLASIT. Each triplet consists of form, start point, and amplitude value.

The number of forms is contained in the only word stored in record 1024.

Common

/CMNDS/

CMAND3 - 1st character of user command

NUM3 - 2nd character of user command

OUTPUT

Common

/DCHEK/TSMOO - status variable of smoothed noise display

CALLED BY KLOOK

SUBROUTINES CALLED NEWVEC

CALLING SEQUENCE CALL DSPIT2

SUBROUTINE NAME DSPIT3

PURPOSE To display the final smoothed approximation to the original data

DISCRIPTION DSPIT3 reads from disk file DPO:FINAL.DAT the final KLASIT results consisting of the smoothed approximation to the original data. DSPIT3 reads the KLASIT waveform sequence, computes x and y differences, and calls NEWVEC to generate the (error corrected) subpicture display. The picture is turned on and assigned the tag and status variable, BFINL. In anticipation of the "merge" user command, a copy of the subpicture is made but not displayed and assigned the tag, TFINL. The picture is left "on" unless the last user command was W or X.

INPUT

Disk

DPO:FINAL.DAT - A FORTRAN direct access file with 1024 records; last record is number of forms and other records are triplets consisting of form, start point, and value.

Common

/CMNDS/

CMAND3 - 1st character of user command

NUM3 - 2nd character of user command

OUTPUT

Common

/DCHEK/

BFINL and TFINL status variables of final smoothed approximations display and its copy, respectively.

CALLED BY KLOOK

SUBROUTINES CALLED NEWVEC

CALLING SEQUENCE CALL DSPIT3

SUBROUTINE NAME DSPRAW

PURPOSE To display the original data.

DESCRIPTION

INPUT

OUTPUT

CALLED BY KLOOK, VU GRAPHICS? Yes

SUBROUTINES CALLED None DISK I/O? No

CALLING SEQUENCE CALL DSPRAW (ARRAY, TRKNUM)

SUBROUTINE NAME DTRMIN

PURPOSE To determine what command (in the smoothing control routine, KLOOK) will produce the current display.

DESCRIPTION DTRMIN is a FORTRAN routine that checks the status of subpictures that can be displayed under control of subroutine KLOOK. Each subpicture is associated with a variable in the /DNAMES/ common block. DTRMIN tests the variables for those that are in the "on" status and returns with CMANDQ and NUNQ arguments set to the characters that would generate the displays that are currently "on".

INPUT Common /DNAMES/ display state variables for subpicture

OUTPUT

Arguments

CMANDQ - The letter ("W", "X", or "D") which will generate the proper display in KLOOK

NUMQ - The accompanying number (1, 2, or 3) completing the command

CALLED BY KLOOK

SUBROUTINES CALLED None

CALLING SEQUENCE CALL DTRMIN (CMANDQ, NUMQ)

SUBROUTINE NAME FATPK

PURPOSE To determine whether a peak found by PEAKPK is too broad.

DESCRIPTION The particular peak which this routine examines is located at LMAX, and the surrounding minimums are at LASMIN and LMIN (with LASMIN < LMIN). The flowchart for FATPK is shown in Figure 3.

Starting at LASMIN and heading to the right toward LMAX, set L1 = the first location (frequency bin) which satisfies one of the following two conditions:

(a) the amplitude at this location exceeds

PERHI * (amplitude at the peak)

or

(b) the amplitude at the location exceeds

PERLO * (amplitude at the peak)

and

$$\frac{\text{(amplitude at this location)}}{\text{(amplitude at previous frequency bin)}} > \text{FACTOR}$$

Note that (a) is a simple amplitude requirement, while (b) has a less restrictive amplitude requirement (since PERLO < PERH1) but adds a condition on the slope or the speed at which the maximum is approached.

If this peak must be split into two peaks, L1 will be the location of the leftmost peak. To locate the peak on the right, start at LMIN and travel to the left toward LMAX; set L2 = the first location satisfying (a) or (b) above.

Having determined the "beginning" (L1) and the "end" (L2) of this peak, compare the width L2-L1+1 with the width needed for a broad peak (ZBROAD). If the peak is broad (>ZBROAD), set SPLIT = .TRUE.; otherwise SPLIT = .FALSE. Return.

SUBROUTINE NAME FATPK (Cont'd)

INPUT

Arguments

ZSPEC - power spectrum array

TPTS - # points in power spectrum

L - current frequency location index

LMAX - location of last minimum

LASMIN - last value of LMIN

OUTPUT

Arguments

L1

- estimated locations of the two peaks into which a "fat peak" should be split

L2

SPLIT - flag set to .TRUE. if the peak in question is too broad

CALLED BY PEAKPK

SUBROUTINES CALLED None

CALLING SEQUENCE CALL FATPK (ZSPEC, TPTS, L, LASMIN, LMAX, LMIN, SPLIT, L1, L2, PERHI, PERLO, FACTOR, ZBROAD)

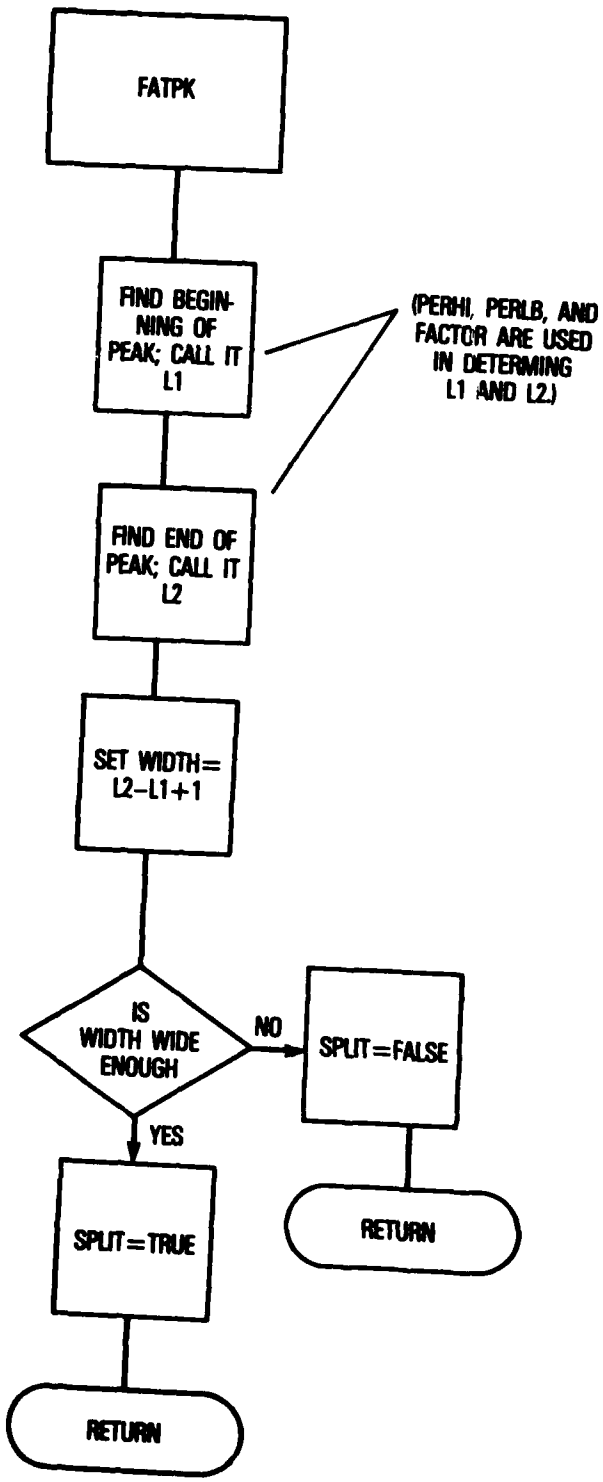


Figure 3 - Subroutine FATPK Flowchart

SUBROUTINE NAME FFTDSP

PURPOSE To calculate and display the real and imaginary parts of the FFT.

DESCRIPTION

INPUT

OUTPUT

CALLED BY VU GRAPHICS? Yes

SUBROUTINES CALLED FOUREA DISK I/O? No

CALLING SEQUENCE CALL FFTDSP

SUBROUTINE NAME FILLIT

PURPOSE To fill ARRAY with PTSDPY points from a specified disk file starting at the point (in the disk file) designated ZFILEP.

DESCRIPTION

INPUT

CALLED BY LOCATE GRAPHICS? No

SUBROUTINES CALLED None DISK I/O? Yes

[User's data file]

CALLING SEQUENCE CALL FILLIT (ICHAN, XFILEP, INCRE, ARRAY)

SUBROUTINE NAME FOUREA [in FOUREA1, MAC]

PURPOSE FOUREA is the Fast Fourier Transform (FFT) routine.

DESCRIPTION

INPUT

OUTPUT

CALLED BY FFTDSP, POCALC GRAPHICS? No

SUBROUTINES CALLED None DISK I/O? No

CALLING SEQUENCE

SUBROUTINE NAME KCMNDS

PURPOSE To operate in conjunction with KLOOK to handle KLASIT level user commands.

DESCRIPTION KCMNDS is a FORTRAN subroutine which turns on, turns off, erases, and checks status of displays specifically associated with the smoothing operations on input data. KCMNDS is called by subroutine KLOOK for the "D", "W", and "M" commands and all forms thereof. KCMNDS checks the current display status and turns on the requested displays if they are not already on or turns them off if they are already on. If the subpicture associated with a requested display does not exist in the display, KCMNDS returns to KLOOK with the control parameter LINE set appropriately to direct KLOOK to call the necessary level of the smoothing process to generate the required display. To direct KLOOK properly, KCMNDS determines the present state of the display by checking the status of the subpictures and erasing undesired displays and displays that must be regenerated.

INPUT

Arguments

CMAND3 - First character of input command

NUM3 - Number following first character of the user command input

Common

/DNAMES/ - list of graphics tags assigned to each subpicture (integers running from 1 to 20)

/DSTAT/ - contains display status array, DCHEK, which records status of each subpicture (on, off, erased)

OUTPUT

Argument

LINE- integer variable containing values from one to eight to indicate branches in KLOOK control structure. Direct KLOOK to various stages of smoothing process.

Common

/DNAMES/ cf. INPUT

/DSTAT/ cf. INPUT

CALLED BY KLOOK

SUBROUTINES CALLED PRAMTR

CALLING SEQUENCE CALL KCMNDS (CMAND3, NUM3, LINE)

SUBROUTINE NAME KINSRT

PURPOSE This routine places a new form into the KLASIT output file (after determining that the form being output is not a duplicate).

DESCRIPTION If the form is the first one (INDEX = 0), KINSRT increases INDEX to 1, sets the 3rd column (KCOL3) equal to the amplitude at the position indicated by the second column (KCOL2), and returns.

If the form is not the first, KTEST is examined to see if it will be necessary to check for a duplicate entry. If KTEST = .TRUE., this form is compared with the previous form, and if they are the same a return is made without increasing INDEX by 1. If KTEST = .FALSE. (which happens only on the first and last two forms), no check is made.

After surviving the duplication check, the form is definitely going to be entered into the output, so INDEX is increased by 1. If the form is a level, it is entered directly into the output, and the previous form is changed so that the third column (the value at the endpoint of the last form) is set to the value of the level. (This modification makes certain that the levels are truly flat and not slightly inclined.) If the form is a rise or fall, the third column is changed by setting it equal to the value at the endpoint (KCOL3 = S(KCOL2)), and then the form is output.

The flowchart for KINSRT is shown in Figure 4.

INPUT

Arguments

S - original data, possibly modified by routine NAROPK; and array dimensioned 1024

INDEX - # of forms already on output file

KCOL1 - first number of form = +1 for rise, -1 fpr fall, 0 for level

KCOL2 - second number of form = endpoint of rise, level, or fall

KCOL3 - last number of form = value at endpoint

KTEST - If KTEST = .TRUE., KINSRT will test to see if a duplicate form is about to be entered.

OUTPUT (DISK) A new entry is sometimes placed on the direct-access file 79 (which may be DPO:UNSMOO.DAT, DPO:SMOO.DAT, or DPO:FINAL.DAT).

CALLED BY KLASIT, LEV, NONLEV

SUBROUTINES CALLED None

CALLING SEQUENCE CALL KINSRT (S, INDEX, KCOL1, KCOL2, KCOL3, KTEST)

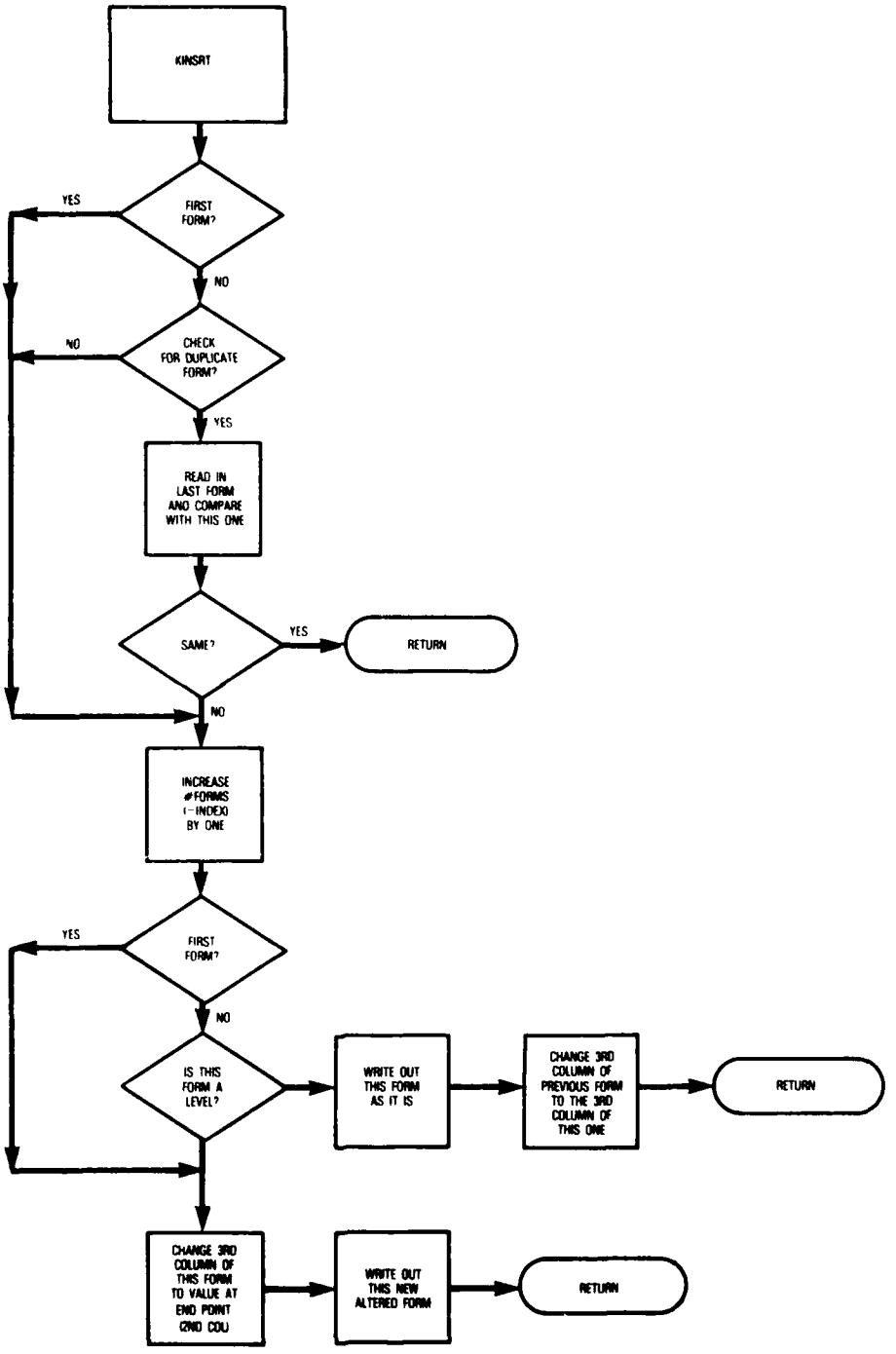


Figure 4 - Subroutine KINSRT Flowchart

SUBROUTINE NAME KLASIT

PURPOSE To characterize a time series of digital data by a sequence of elementary waveforms consisting of rises, falls, and levels.

DESCRIPTION KLASIT reads a series of as many as 1024 values from disk, initializes variables, and cycles through subroutines LEV, NONLEV, NAROPK, and KINSRT to approximate the input data with a sequence of rise, fall, or level waveforms. Each waveform is described by a number triplet and is stored on disk by KINSRT in the order of occurrence. The first number of the triplet is -1, 0, or +1 for fall, level, or rise, respectively. The second number is the index in the original data of the endpoint of the form, and the third number is zero for rise or fall or, for a level, the amplitude of the level. KLASIT works through the raw data input by calling LEV or NONLEV, depending on the possibility of a level form existing at the current data index. Special processing, in the form of a call to NAROPK, is performed to ensure the validity of rises or falls occurring at the end of data.

KLASIT maintains a list in COMMON/PRAMI/ of the variables characterizing the current state of the processed data. These are key values associated with the current waveform indicating its type, bounds, and level. As new forms are encountered, KLASIT maintains a history of the parameter sets for each of the two forms preceding the current form through calls to UPDATE and UPDAT2. This history allows the waveform approximation routines, specifically NAROPK, to eliminate forms and to BACKTRACT and reprocess data to achieve smoother approximations to the input data. The KLASIT flowchart is shown in Figure 5.

INPUT

Arguments

LENGTH - length of the input data

S - input data vector

NABR - vector threshold for level

LEVEL - duration threshold for level

IBAND - constant threshold for significant change in level

P - narrow peak threshold

WHICH1 - parameter indicating the file which is to receive output forms

1. DPO:UNSMOO.DAT
2. DPO:SMOO.DAT
3. DPO:FINAL.DAT

SUBROUTINE NAME KLASIT (Cont'd)

OUTPUT

DISK

DPO:KLSINP.DAT - contains LENGTH and S as they are input to KLASIT

DPO:UNSMOO.DAT -

DPO:SMOO.DAT as indicated by input parameter WHICH1

DPO:FINAL.DAT - contains output sequence of waveforms (a FORTRAN direct-access file containing 1024 three-word records. Record 1024 contains only the number of forms on file)

CALLED BY KLOOK

SUBROUTINES CALLED KINSRT, UPDAT, UPDAT2, LEV, NONLEV, NAROPK

CALLING SEQUENCE CALL KLASIT (LENGTH, S, NABR, LEVEL, IBAND, P, WHICH1.)

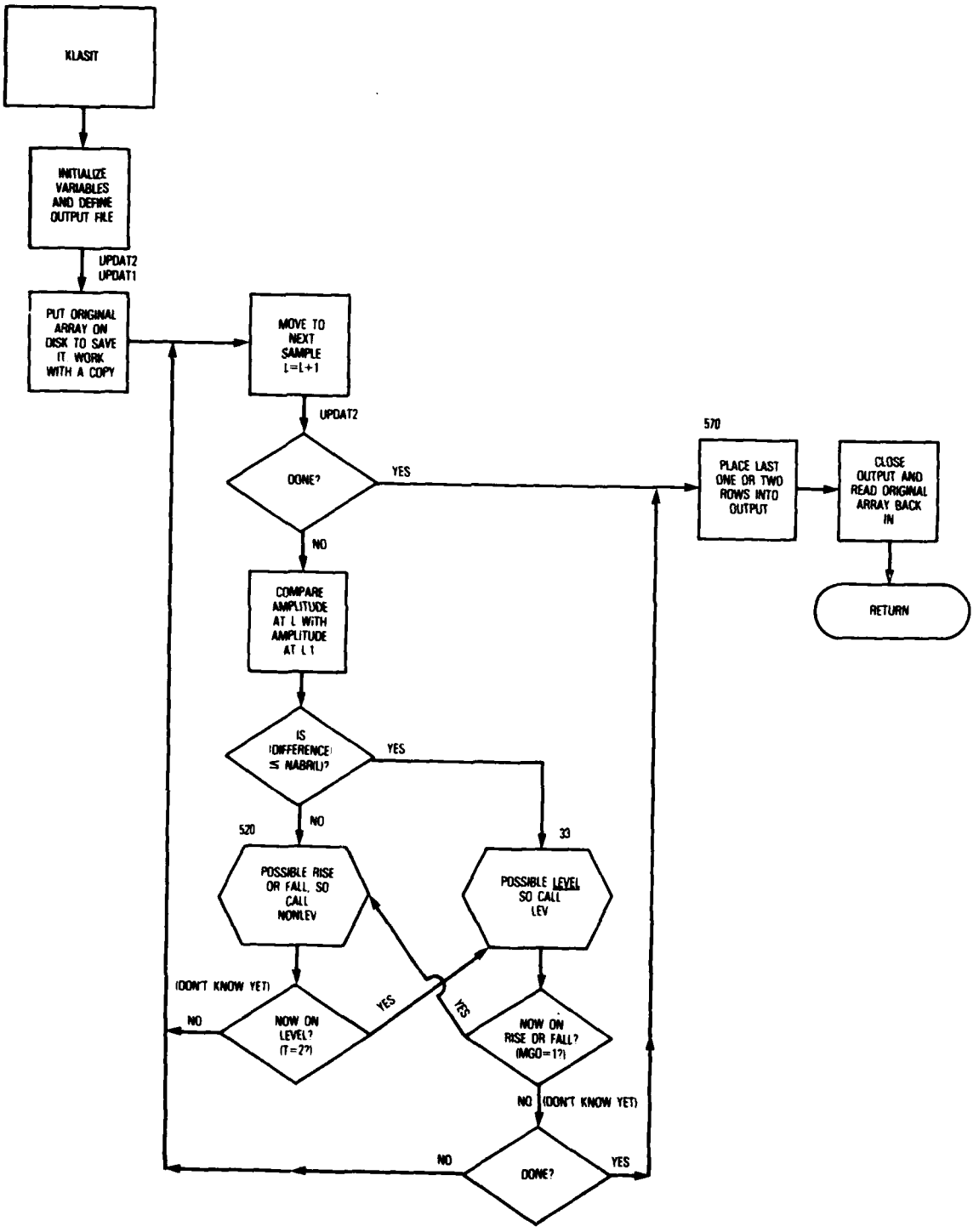


Figure 5 - Subroutine KLASIT Flowchart

SUBROUTINE NAME KLOOK

PURPOSE To control KLASIT smoothing process and associated displays.

DESCRIPTION The FORTRAN subroutine, KLOOK, is the main control routine for the KLASIT smoothing process and related display routines. The user has the option of examining either the original (raw) data or a particular track created by the tracking routines of SELECT. KLOOK accepts all KLASIT commands through calls to QVUCSI and directs program flow accordingly. KLOOK accepts the same data movement commands ("F", "B", "J", "S", and "<CR>") as does VU and moves all displays that are currently on the screen as directed. In addition, the "Q7" and "G" commands in KLOOK have the same effect as they do in VU. The commands peculiar to KLOOK are the "Pn", "Ln", "In", "Nn", and "Qn" comamnds for changing KLASIT parameters and the "Dn" and "Wn" commands for displaying various stages and combinations of raw data, noise, and smoothed signals. All commands eventually cause KLOOK to cycle back to the call to QVUCSI and look for further commands -- the "E" and "K" commands being the means of escaping the KLASIT smoothing module.

When parameters are changed, DTRMIN is called to determine what data are currently displayed and KCMNDS is called to turn off all displays but the original data and set the KLOOK control parameter, LINE, to one or two as needed to effect the proper regeneration of the displays affected by the parameter change.

If a "D" or "W" type command is entered, KCMNDS is called to turn on the requested displays. If the displays exist, KCMNDS returns to KLOOK and KLOOK looks for the next command. If the displays don't exist, KCMNDS returns with the LINE parameter set appropriately to generate the requested displays and KLOOK transfers accordingly. As outlined in the flowchart in Figure 6, KLOOK can cycle through the entire smoothing process including smoothing the raw data using constant noise, computing unsmoothed noise, smoothing the noise, and computing a smoothed signal based on the smoothed noise. KLOOK also affords the only entry to WAMSER which itself is the final process in the smoothing sequence. See Figure 6 for flowchart of KLOOK.

SUBROUTINE NAME KLOOK (Cont'd)

INPUT

Common

/ARRAYS/ARRAY (1024) - original (raw) data

Keyboard

TRKNUM - if equal zero, then examine raw data, otherwise, this is the number of the track that will be smoothed.

OUTPUT None

(Much of the input/output associated with KLOOK is effected by subroutines called by KLOOK. For example, QVUCSI accepts all commands for KLOOK and several display routines generate output.)

CALLED BY VU

SUBROUTINES CALLED CHANGE, CORDSP, CREAT1, CREAT2, DISKRD, DSPIT1, DSPIT2, DSPIT3, DSPRAW, DTRMIN, KCMNDS, KLASIT, PDATA, QVUCSI, TYPEIT, WAMDSP, WAMSER, XYPLOT

CALLING SEQUENCE CALL KLOOK (PDRAW)

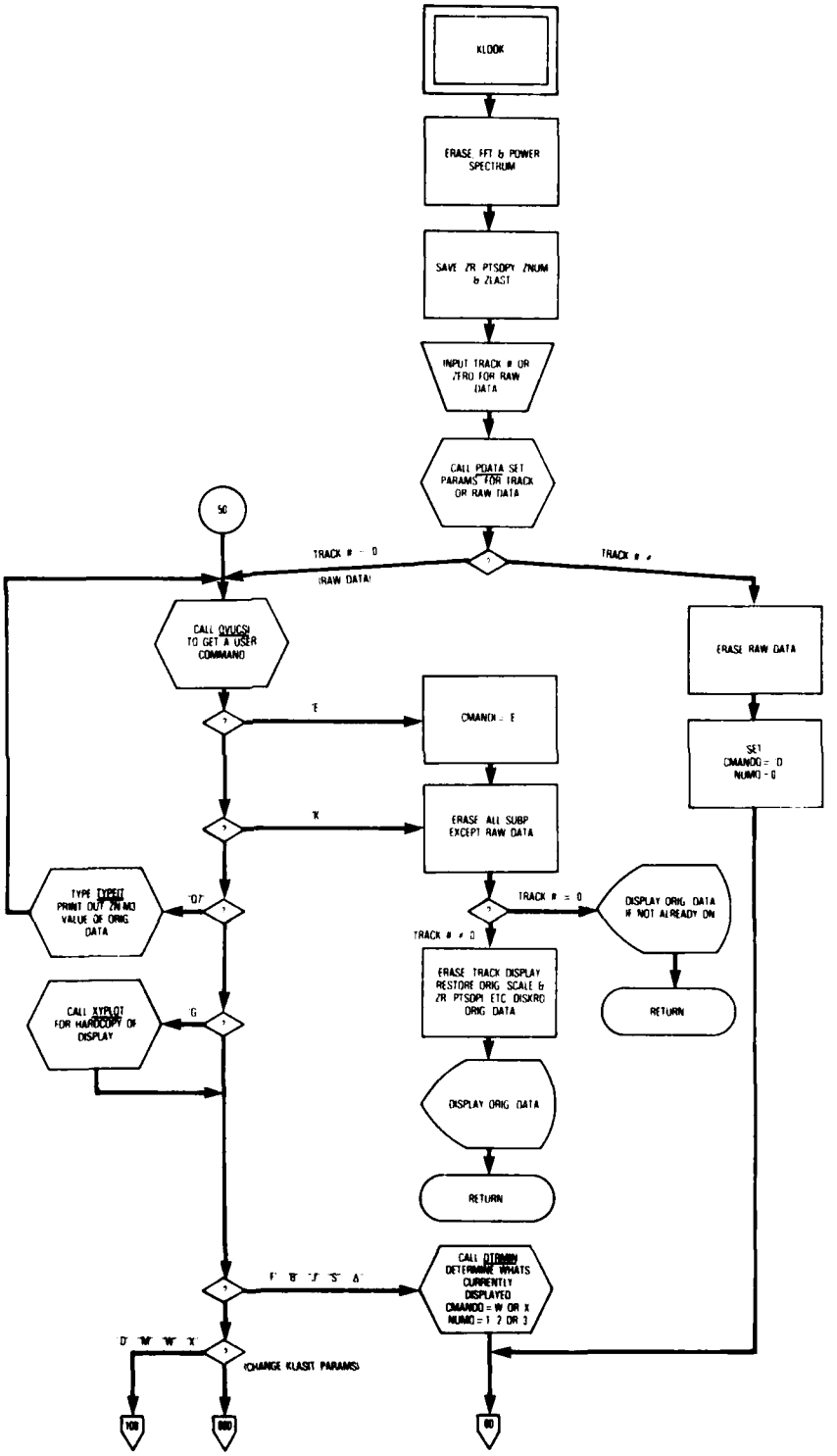


Figure 6 - Subroutine KLOOK Flowchart (1 of 3)

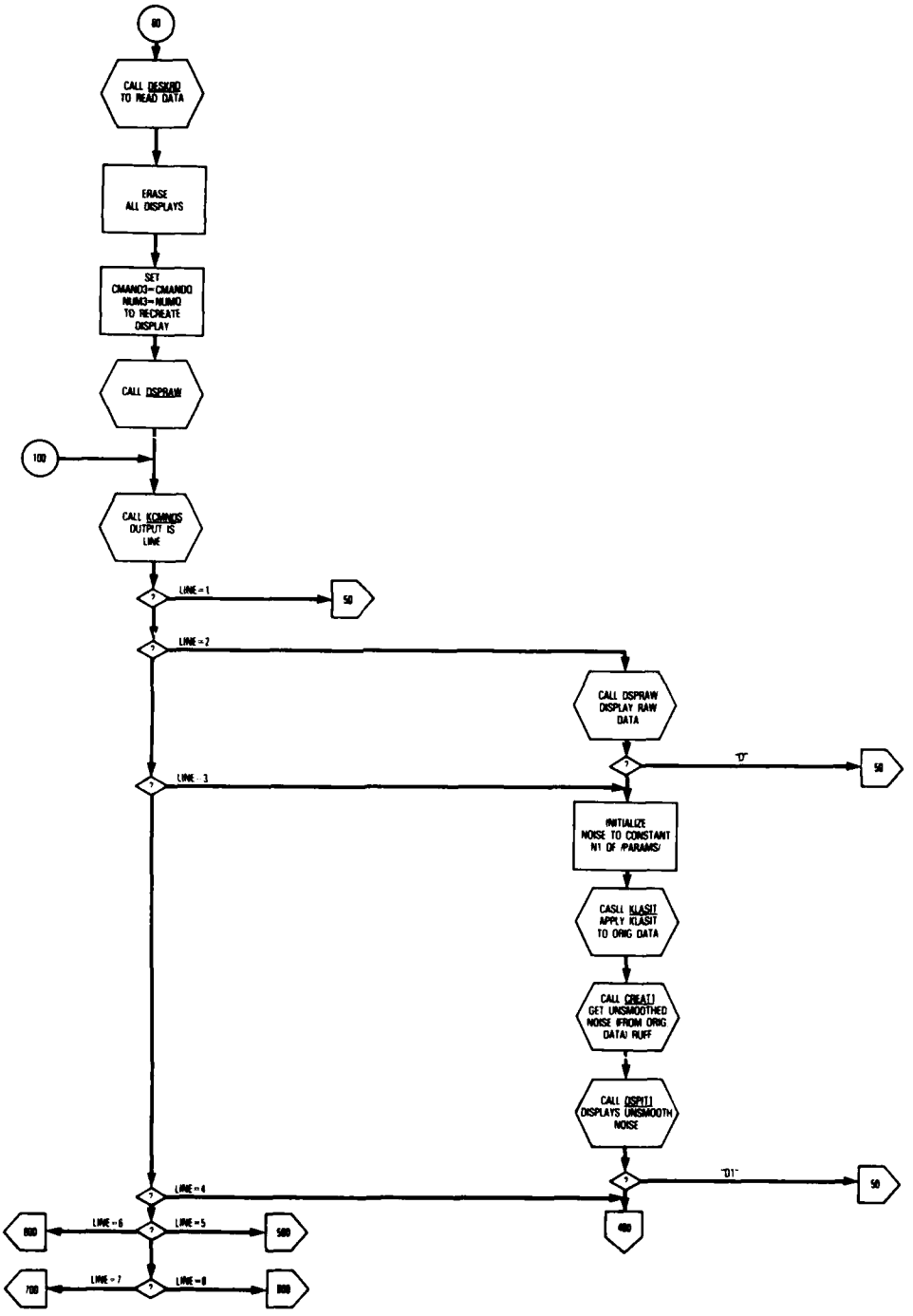


Figure 6 - Subroutine KLOOK Flowchart (2 of 3)

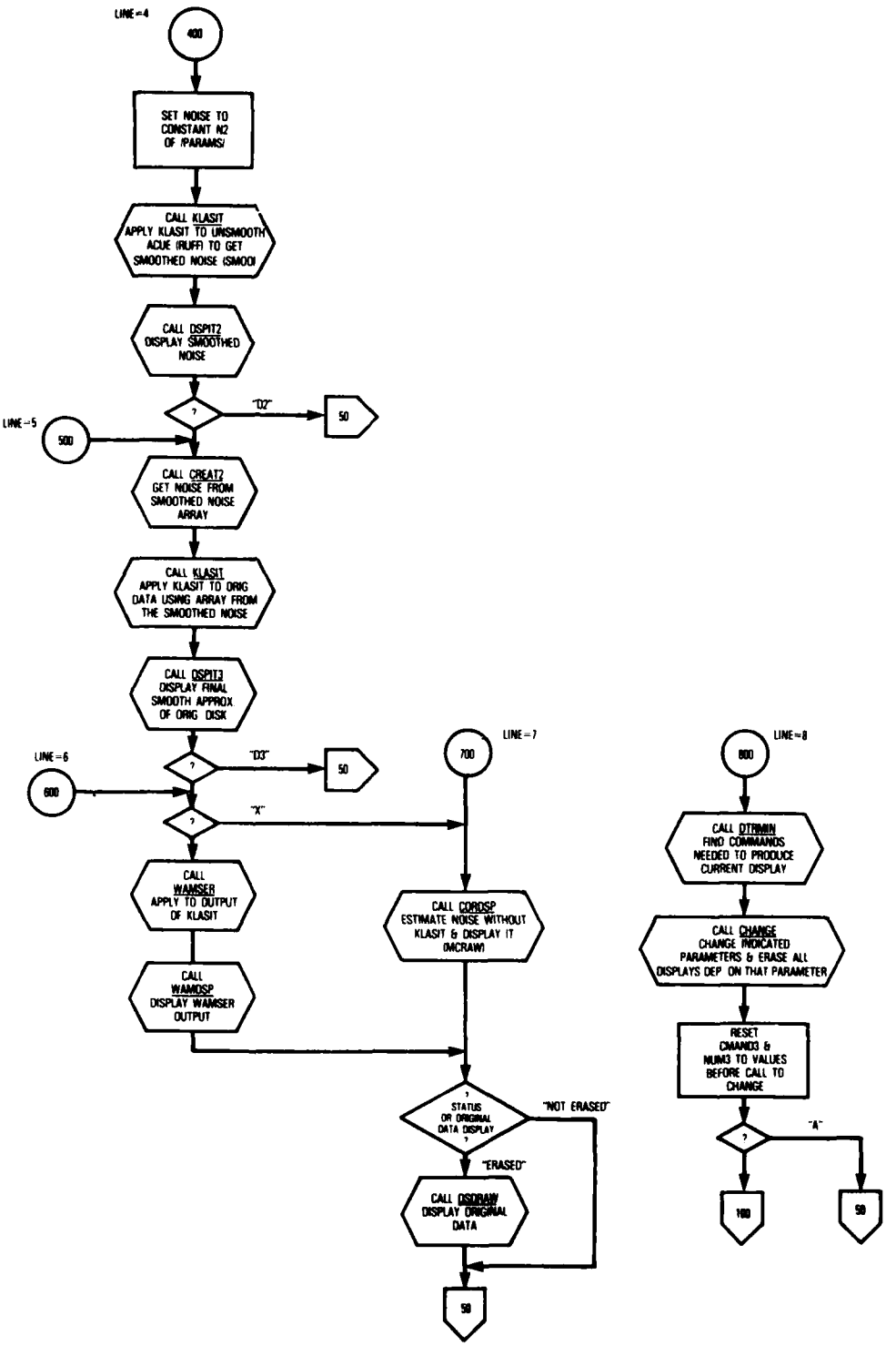


Figure 6 - Subroutine KLOOK Flowchart (3 of 3)

SUBROUTINE NAME KLOUT (KLAS, INDEX)

PURPOSE To output WAMSER results to a disk file for use by the waveform processor.

DESCRIPTION

INPUT

OUTPUT

CALLED BY	None currently	CALLS	None	GRAPHICS?	No
				DISK I/O?	Yes
SUBROUTINES CALLED				[User-specified output file]	
CALLING SEQUENCE					

SUBROUTINE NAME LEV

PURPOSE To test for the existence of a level form and compute its average value and endpoints.

DESCRIPTION Subroutine LEV tests successive elements of the input array FTEMP until a value differs from the value of the start of the level by more than NABR(L), thus signalling the end of the level. If the level width fails to exceed the parameter LEVEL, no level is found and a return is made to process the data as a rise or fall. If the width condition is satisfied, the average value of the level is computed using the original data array, S. The value of the pointer LB to the end of the preceding form is moved to LB1, and LB itself is updated to the start point of this level (L-1) or to the first element in S equal to the average value of the level, whichever comes first. Subroutine NAROPK is called to eliminate narrow peaks if one has just been created, and KINSRT is called to place the level data on disk. The flowchart for subroutine LEV is shown in Figure 7.

INPUT

Arguments

LENGTH - Length of input array to be smoothed

S - 1024-word array containing data to be smoothed

FTEMP - 1024-word array copy of S. This array is analyzed by LEV. (FTEMP contains smoothing modifications. S is never altered to allow the program to restore analysis to pre-existing states.)

NABR - 1024-word vector threshold specifying minimum amplitude change for end of level test

LEVEL - Threshold for minimal width of a level (in number of points)

Common /PRAM1/

L - current pointer to waveform data

SUBROUTINE NAME LEV (Cont'd)

OUTPUT

Arguments:

MGO - status indicator dictating further processing in KLASIT

= 0 means a narrow peak was eliminated and L is currently pointing to a rise or fall

= 1 means level must be absorbed into preceding form

= 2 means end of waveform data - enter last two forms in KLAS

= 3 means a sharp rise after the level at end of waveform data

= 4 means end of data and enter last form in KLAS

Common /PRAM1/

L - points to beginning of next form in data

CALLED BY KLASIT

SUBROUTINES CALLED UPDAT2, NAROPK, KINSRT

CALLING SEQUENCE CALL LEV (LENGTH, S, FTEMP, NABR, LEVEL, IBAND, P, KLAS, INDEX, MGO)

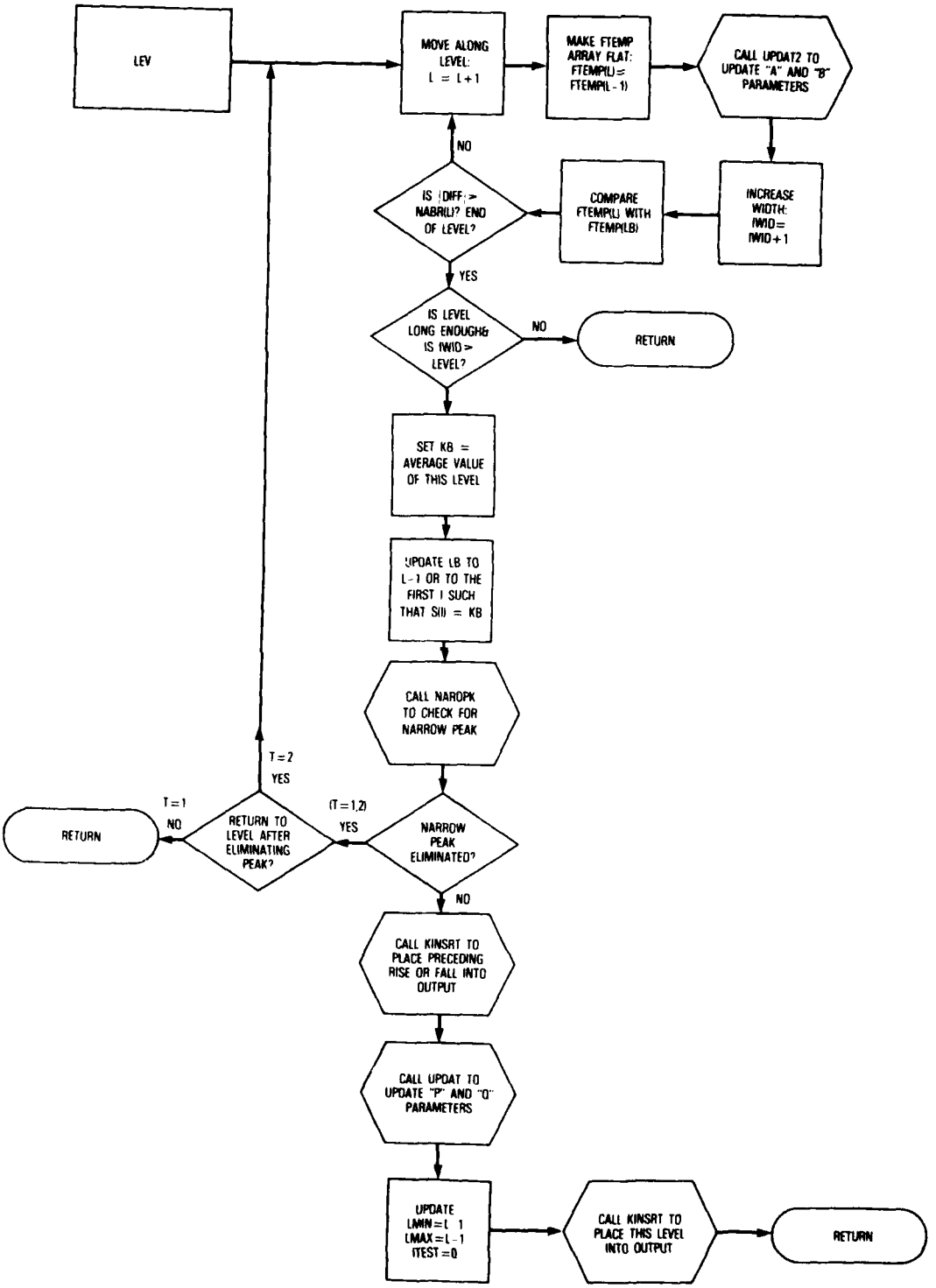


Figure 7 - Subroutine LEV Flowchart

SUBROUTINE NAME LOCATE (ZEVBEG, ZEVEND, ZDURAT)

PURPOSE To locate the beginnings and ends of events.

DESCRIPTION

INPUT

OUTPUT

CALLED BY VU CALLS FILLIT GRAPHICS? No

SUBROUTINES CALLED DISK I/O? Yes

CALLING SEQUENCE [DPO:OPDRAM.DAT]

SUBROUTINE NAME MRGEPK

PURPOSE To merge peaks in PEAKPK if necessary.

DESCRIPTION This routine examines the peak numbered KP. From the output array, the locations of this peak and the previous peak (number KP-1) are determined. If the difference in frequency between these two locations is greater than FMERGE, the peaks are too far away from each other to be merged and a return is made.

If (on the other hand) the peaks are close enough together, they will be merged provided their amplitudes are not too similar. The condition imposed on the amplitudes is that the ratio of the smaller amplitude to the larger amplitude must be less than HMERGE before a merge will be performed. If this condition is not satisfied, a return is made.

If all conditions for a merge are satisfied, the location of the last peak entered is given as the location of the taller peak, and the bandwidth is equal to the sum of the original two. Before returning, the peak number (KP) is reduced by one. The flowchart for MRGEPK is shown in Figure 8.

INPUT (Calling Sequence)

ZSPEC - power spectrum array

TPTS - # points in power spectrum

TMPOUT - output array containing peaks and bandwidths

KP - current peak number

FMERGE - if two peaks are separated by more than FMERGE hertz, they cannot be merged

HMERGE - two candidates for merging are merged if the height of one is less than HMERGE percent of the other

(DISK) -

OUTPUT (Calling Sequence)

KP - peak number is reduced by one if merge occurs

(DISK) -

CALLED BY PEAKPK CALLS None

SUBROUTINES CALLED

CALLING SEQUENCE CALL MRGEPK (ZSPEC, TMPOUT, TPTS, KP, FMERGE, HMERGE)

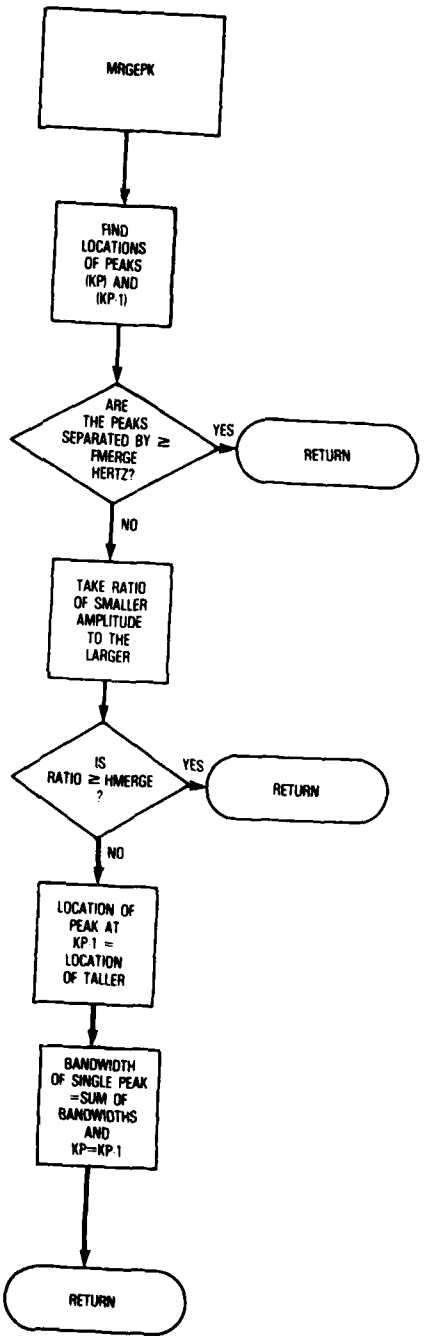


Figure 8 - Subroutine MRGEPK Flowchart

SUBROUTINE NAME NAROPK

PURPOSE To eliminate narrow peaks as part of KLASIT's smoothing process.

DESCRIPTION The routine begins by setting the default condition T=0, meaning that no narrow peak was eliminated. The variable T can take on two other values to tell the calling program what it has done: if T=1, a narrow peak was eliminated and the sample point under consideration is on a rise or a fall; if T=2, a narrow peak was eliminated and the sample point now under consideration is on a level.

A check is then made to see if a peak is really available for examination -- this check is made by multiplying the form types (ITEST) for the current form and the previous form to see if the result is -1; if it is not, one of them is a level, or both are rises, or both are falls.

Once it is determined that the peak is there, the beginning (the endpoint of the last form), the end (LB1), and the width (end-beginning-1) are determined. If the width is >P, the peak is not narrow and a return to the calling routine is made; otherwise the routine continues.

If the peak is narrow, it is eliminated by obtaining amplitude values at the beginning and end of the peak and replacing the intermediate values by a straight-line interpolation between the endpoints. The results are placed in both FTEMP (the working array) and S (the original data). [Note: If S were not changed, calculations of the average level value in the routine LEV would be thrown off, since the S array is used here.]

After zeroing out the last form entered into the output file (which may not be necessary any more), the routine resets L (current sample point) to a point just before the peak began (L = BEGPK-1) and reduces the value of INDEX by 2 (since two forms will be ignored).

If the routine has jumped back to a level, INDEX must be reduced again (by 1), since the rise or fall preceding a level is not entered into the output until the level is.

The internal parameters (LB, LB1, KB, LMAX, LMIN, IWID, LTES, and ITEST) are given the values they had when the routine was at this position earlier in its execution. These old values may be found in the "P" parameters (PLB, PLB1, PKB, PLMAX, PLMIN, PIWID, PLTES, and PITEST). [Note: In the special case where INDEX cannot be reduced by 2 (i.e., at the beginning of the

SUBROUTINE NAME NAROPK (Cont'd)

DESCRIPTION (Cont'd)

waveform, where INDEX <2), these parameters are reinitialized to their values at the start of KLASIT, the final entry in the output is rewritten, and UPDAT is called to reset the "P" and "Q" parameters. A return is then performed with T=1.]

If the routine is now on a level, T is set = 2; otherwise T=1 and a return to the calling routine is made. The flowchart of NAROPK is shown in Figure 9.

INPUT

Arguments

S - original data to be smoothed (possibly altered by previous applications of NAROPK)

FTEMP - copy of the S array (dimensioned, like S, as 1024)

LENGTH - length of waveform to be smoothed (= # elements of S or FTEMP to be used)

P - a peak with width <P is called narrow

KLAS - an array (dimensioned 3) holding information on a single rise, level, or fall

INDEX - # of forms already written to output

Disk - Previous forms are read in from the output file

OUTPUT

Arguments

T - a parameter indicating (a) no narrow peak found (T=0)
or (b) narrow peak eliminated, return to rise or fall (T=1)
or (c) narrow peak eliminated, return to level (T=2)

INDEX - this parameter changes if a narrow peak is eliminated

Disk - The last form entered into the output file may be zeroed, and if the narrow peak is eliminated at the beginning, a new first form is entered

Commons - PRAM1, PRAM2, PRAM3, PRAM4, ZPRAM

CALLED BY KLASIT, LEV, NONLEV

SUBROUTINES CALLED UPDAT

CALLING SEQUENCE CALL NAROPK (LENGTH, S, FTEMP, P, KLAS, INDEX, T)

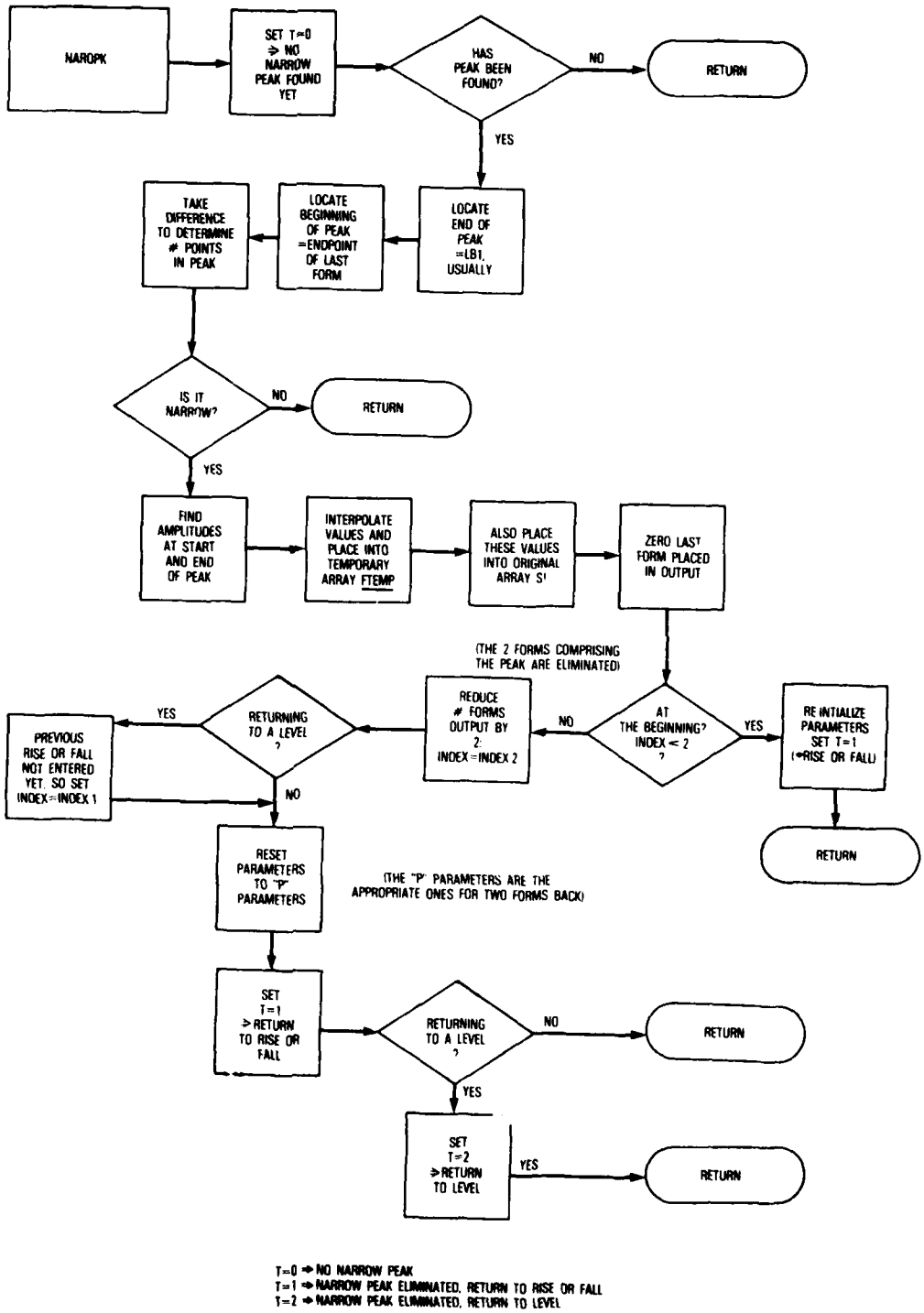


Figure 9 - Subroutine NAROPK Flowchart

SUBROUTINE NAME NEWVEC (XDELTA, YDELTA, XERR, YERR)

PURPOSE To draw a vector with coordinates XDELTA and YDELTA, while keeping track of cumulative errors.

DESCRIPTION

INPUT

OUTPUT

CALLED BY DSPIT1
DSPIT2
DSPIT3
NEXT1 (from PASSBY)
PICTS
WAMDSP

CALLS None

GRAPHICS? Yes

DISK I/O? No

SUBROUTINES CALLED

CALLING SEQUENCE

SUBROUTINE NAME NONLEV

PURPOSE To test for a non-level form (rise or fall) and update extremum points.

DESCRIPTION Subroutine NONLEV compares the wave value at the current wave pointer, L, with the value at the last significant wave pointer, LB, to check for a significant change in amplitude (greater than IBAND + NABR(L)). If there is no significant change, the appropriate local minimum or maximum pointer is updated and NONLEV returns to the calling routine.

If a significant amplitude change occurs (a rise or a fall), LB is backed up to LB1 and reset to L and the appropriate minimum or maximum pointer is updated. In addition, the state variable ITEST is set to -1 for a fall and +1 for a rise. If the preceding form is a level (ITEST=0) or the preceding form is the same as the current form, NONLEV returns to the calling routine. If the current form is a fall preceded by a rise, or vice versa, subroutine NAROPK is called to verify the resulting apparent extremum. If NAROPK eliminates a peak or valley, NONLEV returns without updating ITEST of the max/min pointers. IF NAROPK confirms the validity of the extremum, NONLEV calls subroutine UPDAT, updates the extremum pointer and ITEST, and calls subroutine KINSRT to store the preceding form in the KLAS array. The flowchart for NONLEV is shown in Figure 10.

INPUT

Arguments

S - array containing original data to be smoothed

FTEMP - copy of S array which experiences some modification during smoothing

NABR - vector threshold for significant amplitude changes

IBAND - a single-valued offset added to NABR to determine significant change criterion

LENGTH - length of input S or FTEMP arrays

Common

/PRAM1/ L - current data pointer

LB - pointer to data at last significant change

LMAX - index of last maximum

LMIN - index of last minimum

ITEST - state of last form

-1 - fall

0 - level

1 - rise

OUTPUT

Arguments

T - same as output argument from NAROPK

Common

/PRAM1/ updated L, LB, LMAX, LMIN, ITEST

CALLED BY KLASIT

SUBROUTINES CALLED NAROPK, KINSRT, UPDAT

CALLING SEQUENCE CALL NONLEV (S, FTEMP, KLAS, NABR, IBAND, INDEX, LENGTH, P, T)

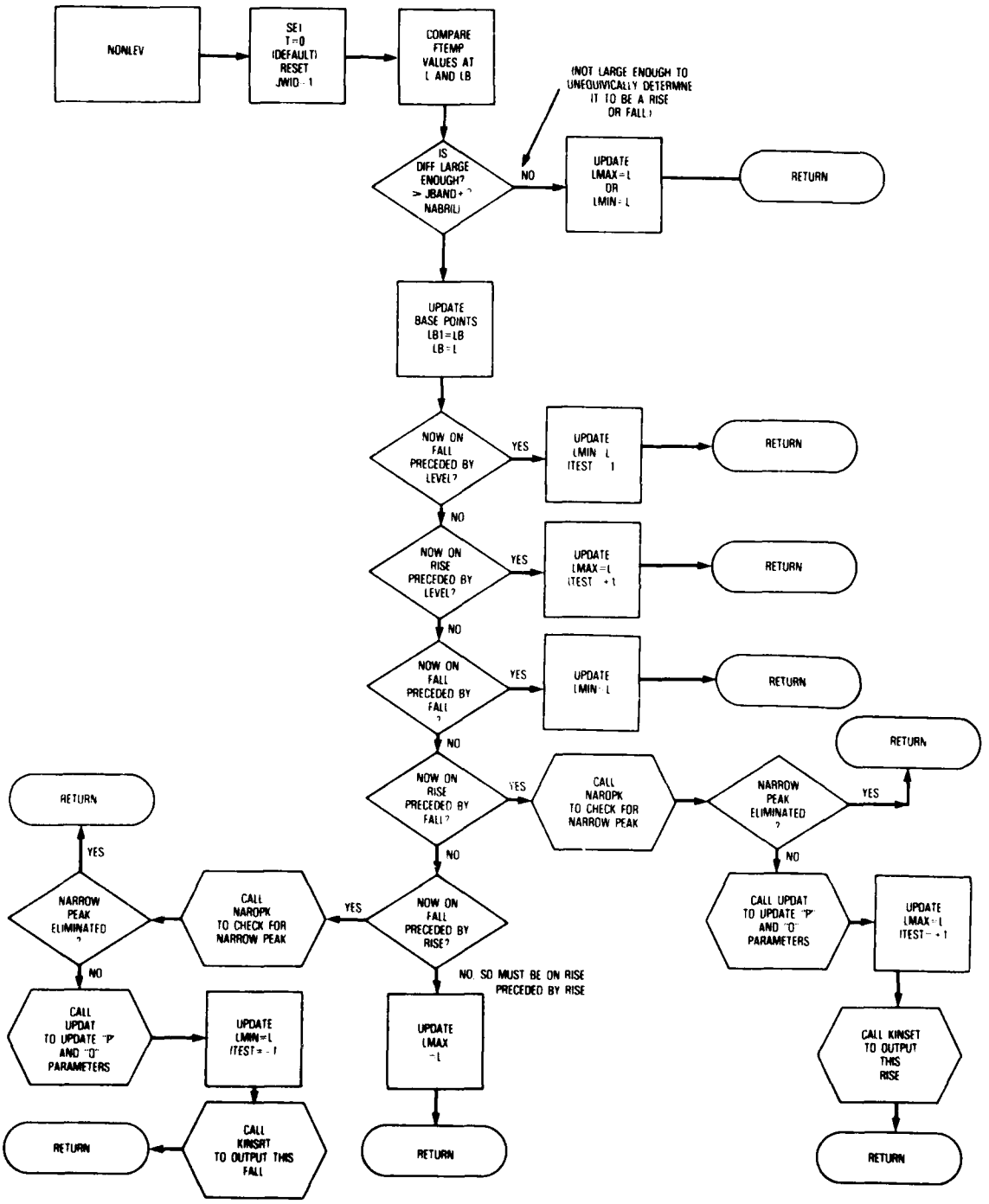


Figure 10 - Subroutine NONLEV Flowchart

SUBROUTINE NAME PDATA

PURPOSE To initialize KLASIT parameters depending on whether tracked data or raw data are being examined.

DESCRIPTION PDATA tests argument TRKNUM and initializes the KLASIT parameters contained in common blocks. For raw data (TRKNUM=0), only parameters in /PARAMS/ are initialized. For tracked data, /PARAMS/ is initialized to different data and also elements of /CHNLS/, /SCALE/, and /CMNDS/ are changed to handle the tracked data.

INPUT

Arguments

TRKNUM - equal to zero to indicate raw data, otherwise, it is set to track number

OUTPUT

Common

/PARAMS/ - new parameter values for KLASIT

/CMNDS/ - new values in case examining tracks

/SCALE/ - new ptsdpy (=50) and other values for tracks

/CHNLS/ - ZLAST = 0 if examining tracks

CALLED BY KLOOK

SUBROUTINES CALLED None

CALLING SEQUENCE CALL PDATA (TRKNUM)

SUBROUTINE NAME PEAKPK

PURPOSE To select peaks in a power spectrum.

DESCRIPTION The routine begins by reading in the parameters to be used. They are as follows:

(1) AMPMIN - For a point to be recognized as a peak, it must exceed in amplitude (AMPMIN percent of) the average value of the power spectrum.

(2) BAND - This parameter determines what is called a significant change; a significant change must be greater than

$$\frac{\text{(value at this index)} + \text{(value at last base)}}{2} * \text{BAND}$$

(3) PERHI - This parameter is the percentage of the maximum (peak value) required for the start or end of a peak.

(4) PERLO - This number is the percent of the maximum required before a slope comparison with FACTOR can be made.

(5) FACTOR - This number is the minimum slope required for the start or end of a peak if (PERLO * maximum) is exceeded.

(6) BWIDTH - A peak must be at least BWIDTH hertz wide before it is resolved into two peaks.

(7) SFREQ - Peakpicking will begin at frequency SFREQ.

(8) FMERGE - If two peaks are separated by less than FMERGE hertz, they are candidates for merging.

(9) HMERGE - Two candidates for a merge are merged if the height of one is less than HMERGE percent of the other.

After reading in these parameter values, BWIDTH is immediately converted to its equivalent in power spectrum sample points (called ZBROAD). After initializing some variables, the average value of the entire power spectrum is calculated and saved for later use. The starting frequency bin L is determined from the parameter SFREQ, and the algorithm is ready to begin.

The power spectrum is scanned from left to right starting at the frequency bin corresponding to SFREQ. The frequency bin index L is repeatedly updated by 1 as the algorithm proceeds; this index is trailed by another index (LB) - the "last base" - which is updated to L every time a change of direction is

SUBROUTINE NAME PEAKPK (Cont'd)

DESCRITPION (CONT'd)

encountered. At each step, values of the last maximum (LMAX) and last minimum (LMIN) are available, and a third variable (ITEST) keeps track of whether the index L-1 is on a rise (ITEST = 1) or a fall (ITEST = -1).

At each step, it is determined if a "significant change" has occurred. This significant change is defined by the requirement that

$$\text{/difference between values at L and LB/} > \begin{matrix}\text{average of values} \\ \text{at L and LB}\end{matrix} * \text{BAND}$$

Incidentally, a little algebra shows that this condition is equivalent to

$$\text{/log (value at L) - log (value at LB)/} > \log \frac{2 + \text{BAND}}{2 - \text{BAND}}$$

If no significant change is encountered, the routine moves on, updating LMAX and LMIN. If a significant fall occurs, LMIN is updated, ITEST is set to -1, and the routine moves on. If a significant rise occurs, LMAX is updated, ITEST is set to +1, and the routine may or may not move on. It moves on along the power spectrum only if the last direction of the power spectrum was also a rise; if it was a fall, then a minimum has been found.

This routine locates a peak by finding the minimum values on either side of it.

Once a minimum is found, the routine decides that between this minimum (LMIN) and the last minimum (LASMIN = last value of LMIN) there is a peak (at LMAX) to be examined. If this peak is not high enough - meaning its amplitude must be > AMPMIN percent of the average value of the power spectrum -- then it is ignored and the routine proceeds as before.

If the peak is high enough, FATPK is called to determine if the peak is broad enough to warrant being split into two separate peaks (PERHI, PERLO, FACTOR, and ZBROAD are used by FATPK). If the peak is split, both peaks are entered into the output array, the appropriate parameters are updated, and the program continues as before.

If the peak is not broad enough to be split, the subroutine MRGEPK (which uses HMERGE and FMERGE) is called to determine if it should be merged with a previous peak, and to "merge it" in the output array if necessary. If no merger occurs, the single peak is placed in the output array.

SUBROUTINE NAME PEAKPK (Cont'd)

DESCRITPION (CONT'd)

The routine continues in this way until 1) the end of the power spectrum is encountered, or 2) the specified number of peaks (=NOP $\leq$ 10) is found. The flowchart for PEAKPK is shown in Figure 12.

NOTE ON BANDWIDTHS: Currently PEAKPK determines a bandwidth for each peak and places it in the array TMPOUT along with the location of the peak. However, bandwidth is not used at the moment and so only the location information in TMPOUT is sent back from PEAKPK.

Here's how the bandwidths are calculated: the parameter PABOVE (not user-controlled at this time) determines the value of the bandwidth, depending on what the program has calculated for the beginning and the end of the peak.

Instead of taking B to represent the bandwidth of the peak in Figure 11a, the shaded triangle shown in figure 11b can be divided into two parts by a solid line in such a way that the area in the triangle above the dark line divided by the total area of the triangle is PABOVE.

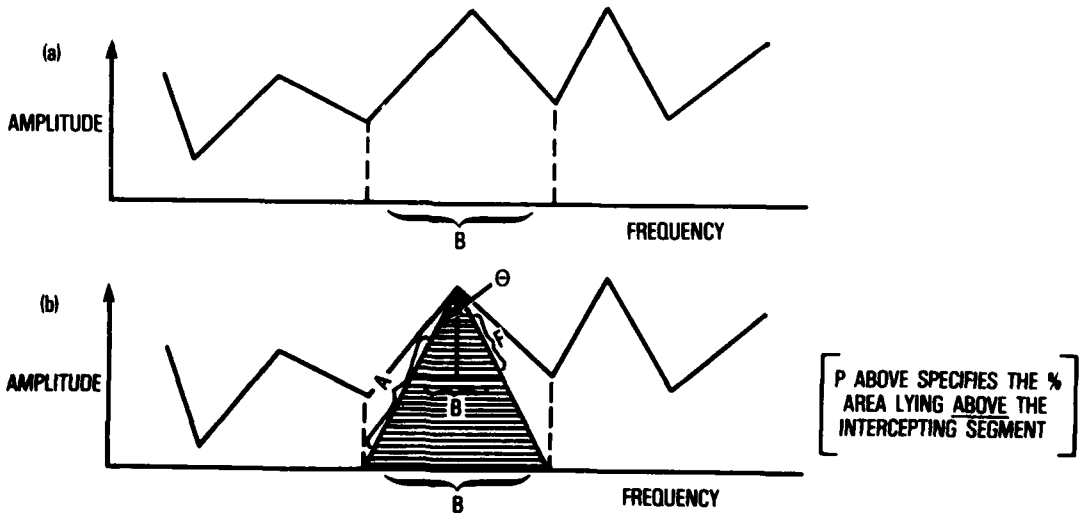


Figure 11 - Peak Bandwidth

SUBROUTINE NAME PEAKPK (Cont'd)

DESCRIPTION (Cont'd)

Since $\frac{\text{small area}}{\text{large area}} = \frac{1/2\ ab \sin \theta}{1/2\ AB \sin \theta} = \frac{a}{A}\ \frac{b}{B} = \frac{b}{B}^2$ = PABOVE, the bandwidth b is defined by b = B PABOVE. This formula is used to calculate bandwidths for all the peaks found by PEAKPK, but the information remains in PEAKPK and is not used.

INPUT (Calling Sequence)

ZSPEC - power spectrum array (dimensioned 512)

TPTS - number of points in this power spectrum

(DISK)

Appropriate parameters (BAND, PERHI, PERLO, FACTOR, FMERGE, HMERGE, BWIDTH) are read in from DPO:OPARAM.DAT.

OUTPUT (Calling Sequence)

JNKOUT - output array (dimensioned 10)

(DISK) -

Commons: Scale

CALLED BY POWDSP, SEARCH CALLS: FATPK, MRGEPK

SUBROUTINES CALLED

CALLING SEQUENCE CALL PEAKPK (ZSPEC, TPTS, JNKOUT)

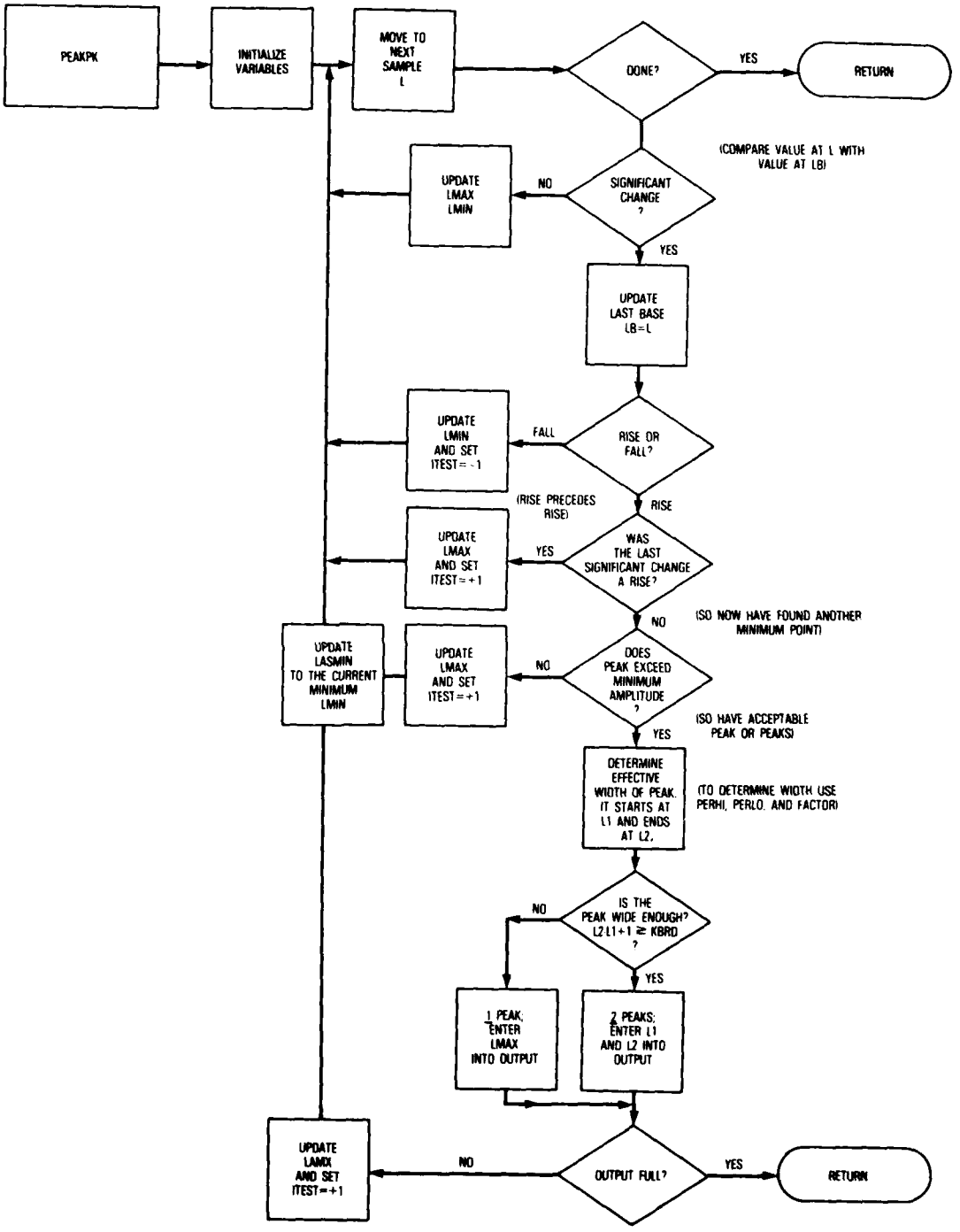


Figure 12 - Subroutine PEAKPK Flowchart

SUBROUTINE NAME PICTS

PURPOSE To set up some standard (straight-line) pictures to be used repeatedly by VU routines.

DESCRIPTION

INPUT

OUTPUT

CALLED BY	VU	CALLS	NEWVEC	GRAPHICS?	Yes
SUBROUTINES CALLED				DISK I/O?	No

CALLING SEQUENCE

SUBROUTINE NAME PKSUB (TMPOUT, ZSPEC, POWPTS, LOGPOW, NOP)

PURPOSE To display the peakpicker results (flashing X's) on top of the power spectrum display.

DESCRIPTION

INPUT

OUTPUT

CALLED BY	POWDSP	CALLS	None	GRAPHICS?	Yes
				DISK I/O?	No

SUBROUTINES CALLED

CALLING SEQUENCE

SUBROUTINE NAME POCALC (PTSDPY, ZPSCAL, LOGPOW)

PURPOSE To calculate the power spectrum, low power spectrum, or phase spectrum (as determined by LOGPOW).

DESCRIPTION

INPUT

OUTPUT

CALLED BY	POWDSP	CALLS	BESSEL	GRAPHICS?	No
	SEARCH		FOUREA	DISK I/O?	Yes
			STRA10	[DPO:WINDOW.DAT]	

SUBROUTINES CALLED

CALLING SEQUENCE

SUBROUTINE NAME POWDSP (PDRAW)

PURPOSE To display the power spectrum (or log power spectrum or phase spectrum) with peakpicker results superimposed if desired.

DESCRIPTION

INPUT

OUTPUT

CALLED BY	VU	CALLS	POCALC, POWSUB	GRAPHICS?	Yes
SUBROUTINES CALLED			PEAKPK, PKSUB	DISK I/O?	Yes
CALLING SEQUENCE			QVUCSI	[DPO:OPARAM.DAT]	

SUBROUTINE NAME POWSUB (LOGPOW, ZSPEC, POWPTS)

PURPOSE To display the contents of ZSPEC (usually the power spectrum) for POWDSP.

DESCRIPTION

INPUT

OUTPUT

CALLED BY	POWDSP	CALLS	NEWVEC	GRAPHICS?	Yes
SUBROUTINES CALLED				DISK I/O?	No
CALLING SEQUENCE					

SUBROUTINE NAME PRAMTR

PURPOSE To display values of specified parameter groups in common /PARAMS/.

DESCRIPTION

The common block /PARAMS/ contains three groups of program parameters:

1) N1, L1, I1, P1, Q0, Q1

2) N2, L2, I2, P2

3) L3, I3, P3

Subroutine PRAMTR displays the group of parameters and their current values specified by the value of the argument WHICH. The display generated is assigned the subpicture number PR1, PR2, or PR3, corresponding to the group number, and DCHEK (PRi) is set to ON to record which subpicture is displayed.

INPUT

Argument

WHICH - is equal to one, two, or three depending on which parameter group is to be displayed

Common variables:

PR1, PR2, PR3 in /DNAMES/ contain subpicture tag numbers set aside for each group ($0 < PRi \leq S20$).

DCHEK(20) in /DSTAT/ - DCHEK (PRi) is set to ON if group i displayed

OUTPUT None

CALLED BY KCMNDS

SUBROUTINES CALLED None

CALLING SEQUENCE CALL PRAMTR (WHICH)

SUBROUTINE NAME PRESS

PURPOSE To accept an array of ASCII characters and move all the blanks in the array to the right.

DESCRIPTION The input line INPUT is scanned from left to right by two indices, the first (I) always trailing the second (J). Whenever J encounters a non-blank character, this character is placed into INPUT (I), and a blank is placed into INPUT (J). This press continues until J > 51, at which time all blanks will have migrated to the right. The flowchart for PRESS is shown in Figure 13.

INPUT

Argument

INPUT - a logical array (dimensioned 51) containing input line

OUTPUT

Argument

INPUT - same array with all blanks pushed to the right

CALLED BY QVUCSI

SUBROUTINES CALLED None

CALLING SEQUENCE CALL PRESS (INPUT)

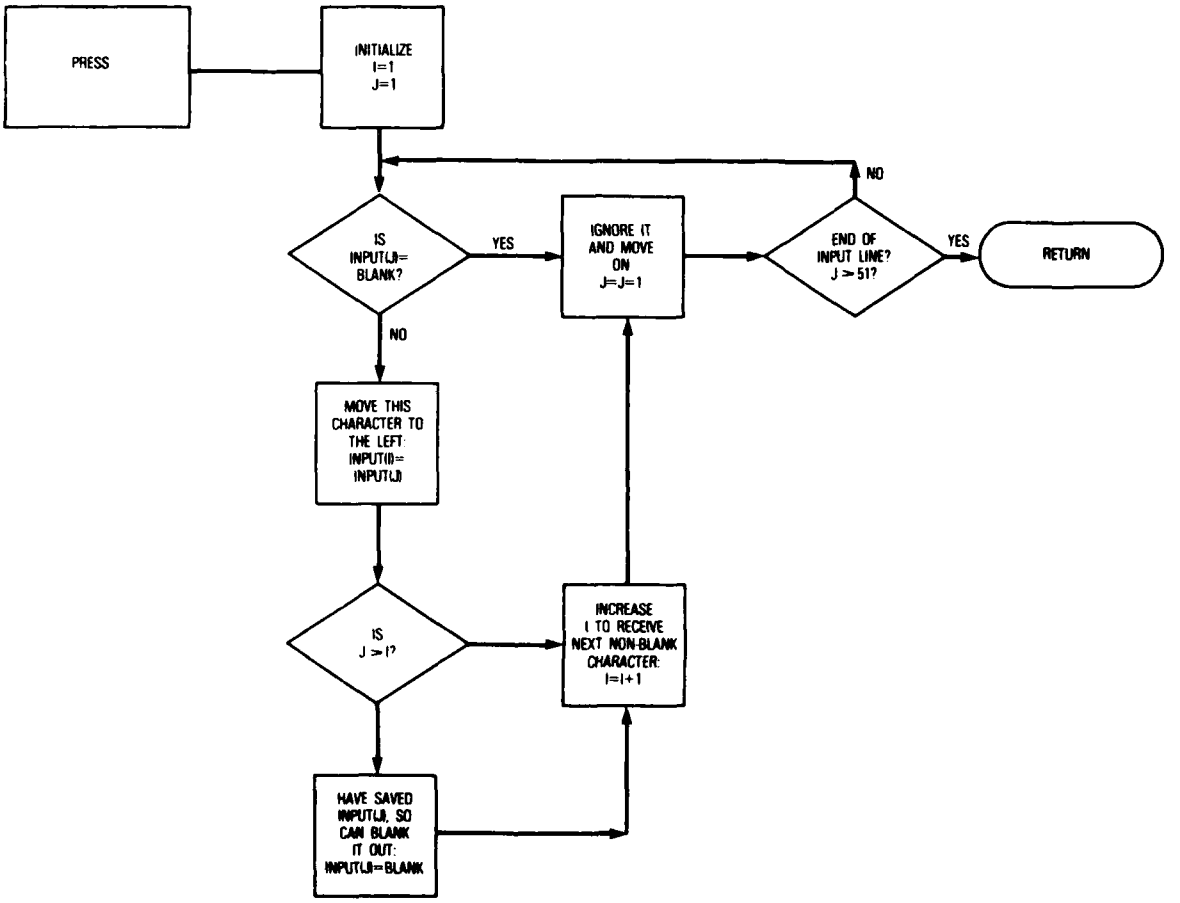


Figure 13 - Subroutine PRESS Flowchart

SUBROUTINE NAME QVUCSI

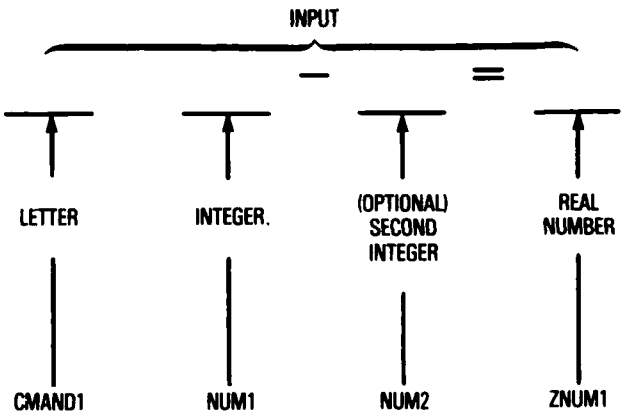
PURPOSE To act as a command string interpreter by converting an ASCII string into codes recognizable by the VU routines.

DESCRIPTION This routine begins by initializing those (logical) strings which will be used to hold intermediate results. These strings are CMAND, CNUM1, CNUM2, and CNUM3. They will give rise to the output as follows:

CMAND will produce CMAND 1 (a single letter)
CNUM1 will produce NUM1 (an integer)
CNUM2 will produce NUM2 (an integer)
CNUM3 will produce ZNUM1 (a real number)

NUM3, although listed as an output variable, is currently not needed.

Both VU and SELECT were originally designed to accept commands in the following form:



SUBROUTINE NAME QVUCSI (Cont'd)

DESCRIPTION (Cont'd)

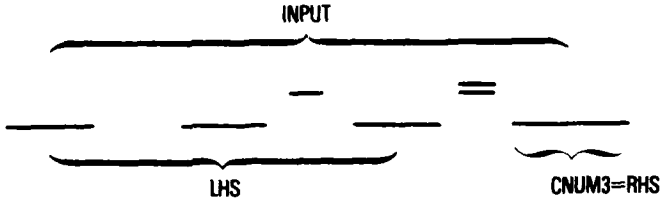
QVUCSI was designed to allow more flexibility in input (for example, instead of using P for power spectrum, the entire word POWER SPECTRUM can be typed for clarity) and to allow merger of the VU and SELECT programs. It was thought that eventually such cryptic commands as P5 (apply PEAKPK to power spectrum and display the results) could be changed to a more readable and more easily remembered form.

QVUCSI receives the string INPUT to be processed, and the first objective is to locate the equals sign, which will divide the input line into two "halves," the left-hand (LHS) and the right-hand (RHS) sides.

PRESS (documented elsewhere) is called first to move all blanks to the right in INPUT, and applying the system routine SCOPY (see the system subroutine manual for a description of all routines other than PRESS used by QVUCSI) puts a null byte (=zero) at the end of INPUT. With the null byte, INPUT can be manipulated by other system subroutines such as TRIM, which is now used to chop off trailing blanks.

A call to INDEX puts the location of the equals sign into the variable EQLOC. If no equals sign is found, INDEX sets EQLOC =0; in this case the equals sign should be assumed at the far right of INPUT, so EQLOC is reset to one more than the (non-zero) length of INPUT (EQLOC - len (INPUT) + 1).

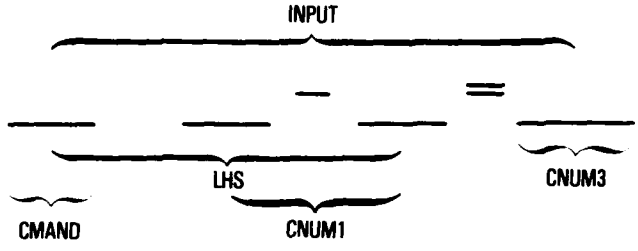
Next SUBSTR is called to put the RHS into the string called CNUM3. The input has now been subdivided as follows:



SUBROUTINE NAME QVUCSI (Cont'd)

DESCRIPTION (Cont'd)

Next VERIFY is used in a loop to find the place on the LHS where the numbers begin. Then SUBSTR is called twice to divide the LHS into two strings (a number string and a letter string) as follows:



As shown, the letter string is called CMAND (and may consist of more than one letter at the moment) and the number string is called CNUM1 (which may consist of two integers separated by a dash). Special case: If no numbers at all appear on the LHS, the string CNUM1 is defined to be the string which starts with the equals sign and continues to the right, including the RHS in the process. This step is somewhat clumsy, since CNUM1 should be the null string at this point, but for "historical reasons" it was not done right away. This CNUM1 array does become null a few steps later (because it begins with an equals sign), so no harm is done.

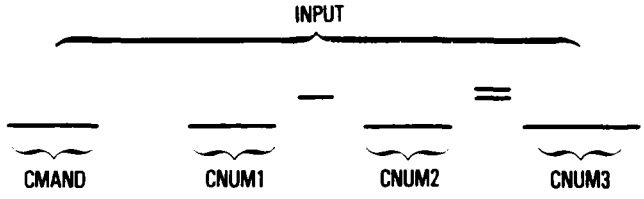
Since the CNUM1 array might be two numbers, VERIFY is called to locate the separator between them (usually a dash). Then CNUM2 is truncated at this

SUBROUTINE NAME QVUCSI (Cont'd)

DESCRIPTION (Cont'd)

separator by SCOPY and the removed part is placed into the string CNUM2. (Note that the truncation of CNUM1 yields the null string if CNUM1 begins with an equals sign, as it does in the special case mentioned above.)

The input has now been divided into substrings as shown below:



From these four strings the appropriate number can be determined (with the help of the DECODE facility). Using DECODE, CNUM1 yields NUM1, CNUM2 yields NUM2, and CNUM3 yields ZNUM1. CMAND1 is defined as the first element of the string CMAND. (If CMAND is the null string, CMAND1 is defined as a blank.) The flowchart for QVUCSI is shown in Figure 14.

Note: Since CMAND1 is given its value by setting it equal to a logical variable, the high-order byte is filled with zeroes. However, CMAND1 when used by VU (and SELECT, maybe) is compared with words having the ASCII code for a blank in the high-order byte, so this code must be placed into CMAND1 by doing an ".OR." operation as follows:

```
CMAND1 = CMAND1 .OR. 20000
```

INPUT (Calling Sequence)

INPUT - a logical array (dimensioned 51) containing up to 50 ASCII characters

(DISK)

OUTPUT (Calling Sequence)

CMAND1 - integer variable containing code letter

NUM1 - integer variable containing code number

NUM2 - integer variable containing code number

NUM3 - integer variable containing code number

ZNUM1 - real variable containing code number

(DISK)

Commons None

CALLED BY VU, KLOOK, POWDSP CALLS PRESS

SUBROTUINES CALLED

CALLING SEQUENCE CALL QVUCSI (INPUT, CMAND1, NUM1, NUM2, NUM3, ZNUM1)

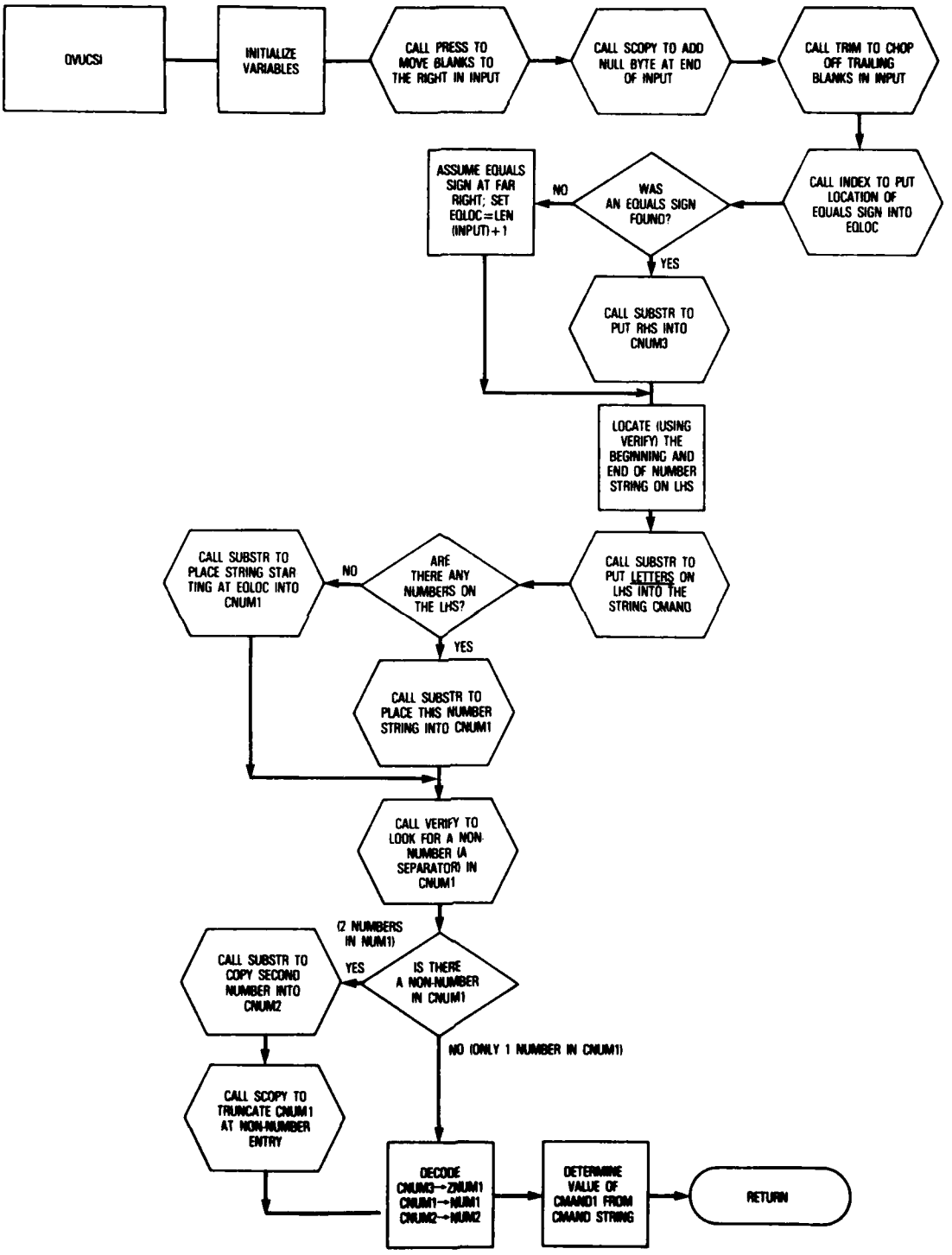


Figure 14 - Subroutine QVUCSI Flowchart

SUBROUTINE NAME REEDTR (DSKBUF, IBLK, TRKNUM, ICHAN)

PURPOSE To read the disk for DISKRD -- 256 words from block # IBLK into the array DSKBUF using channel ICHAN.

DESCRIPTION

INPUT

OUTPUT

CALLED BY DISKRD CALLS None GRAPHICS? No

SUBROUTINES CALLED DISK I/O? Yes

CALLING SEQUENCE [DPO:TRAKS2.DAT]

SUBROUTINE NAME SAVEIT (OUTNUM, RAW)

PURPOSE To store up to five display buffers on disk (to be read in and displayed by the separate program MANYVU).

DISCRIPTION

INPUT

OUTPUT

CALLED BY VU CALLS None GRAPHICS? Yes

SUBRTOUTINES CALLED DISK I/O? Yes

CALLING SEQUENCE [DPO:TEST00.DPY
DPO:TEST01.DPY
DPO:TEST02.DPY
DPO:TEST03.DPY
DPO:TEST04.DPY]

SUBROUTINE NAME SEARCH

PURPOSE To determine major peaks in consecutive power spectra and place the results in a file called DPO:MRPK2.DAT., input for the SELECT program.

DESCRIPTION This routine begins by reading in appropriate parameters from the file DPO:OPARAM.DAT, namely

SFREQ - the starting frequency for the peakpicker

LOGPOW - a parameter which currently specifies both the type of window (Kaiser-Bessel or rectangualr) and the type of spectrum (power, log-power, or phase)

ZPSCAL - power spectrum scale factor

Next the output file is set up (DPO:MORPK2.DAT) and the bin number (LSTRT) corresponding to SFREQ is calculated.

The user is now prompted to specify

(1) where the first spectrum will be taken (ZSTART)

(2) the amount to be shifted before the next spectrum is taken (ZMOVE)

(3) the total number of spectra to be taken (ZWINDS).

These three values are written into the last record (record #1024) of the output file along with the number of points of data submitted to each FFT (PTSDPY).

The current sample number (ZNUM) is saved so that the array now being displayed can be replaced after this routine is done.

Now the routine calculates NWINDS power spectra by successively repeating the following steps:

(1) DISKRD is called to shift by ZMOVE samples and fill the array ARRAY with raw data from this new vantage point

(2) An array PPOUT is initialized to zero

(3) POCAL is called to calculate the spectrum for these data (the data in ARRAY)

(4) PEAKPK is called to pick out prominent peaks in the spectrum

(5) The results of PEAKPK are placed into the PPOUT array after removing the offset (STRT) corresponding to SFREQ.

WARNING: LSTRT was originally subtracted so that peaks picked high in the spectrum could be easily displayed when the SEARCH output was displayed within the VU programs. Now that an entirely new set of routines is

SUBROUTINE NAME SEARCH (Cont'd)

DESCRIPTION (Cont'd)

available for the display and tracking of the SEARCH output, it is best while using SEARCH to keep LSTRT=1 (or equivalently SFREQ=0). Otherwise frequency readings in SELECT will not be correct, since the starting frequency of the peakpicker (SFREQ) is not placed in the output file DPO:MORPK2.DAT to be used by SELECT in determining frequencies. Since SELECT can display and track in any area of the spectrum, it is best to keep SFREQ=0 while performing SEARCH and perhaps to increase the number of peaks sought by PEAKPK to catch the higher frequencies.

(6) The PPOUT values are written onto a single record in the output file, along with a variable LTYPE (set identically = 1 in this routine) which will be used by SELECT to "mark" or "unmark" spectral slices.

After these six steps have been performed NWINDS times, SEARCH closes the output file and has VOTRAX announce that the peakpicking has been completed.

The flowchart for SEARCH is shown in Figure 15.

INPUT (Calling Sequence)

(DISK)

Reads in the parameters SFREQ, starting frequency, LOGPOW, indicator of spectrum type, and ZPSCAL, power spectrum scale factor from the file DPO:OPARAM.DAT.

OUTPUT (Calling Sequence)

(DISK)

Puts peakpicking results for all the power spectra taken on the file DPO:MORPK2.DAT.

Commons:	ARRAYS	DNAMES	
	DSTAT	CMNDS	
	LETRS	CHUNKS	
	SCALE		
CALLED BY	VU	CALLS	DISKRD, POCALC,
SUBROUTINES CALLED			PEAKPK, SHOUT

CALLING SEQUENCE CALL SEARCH

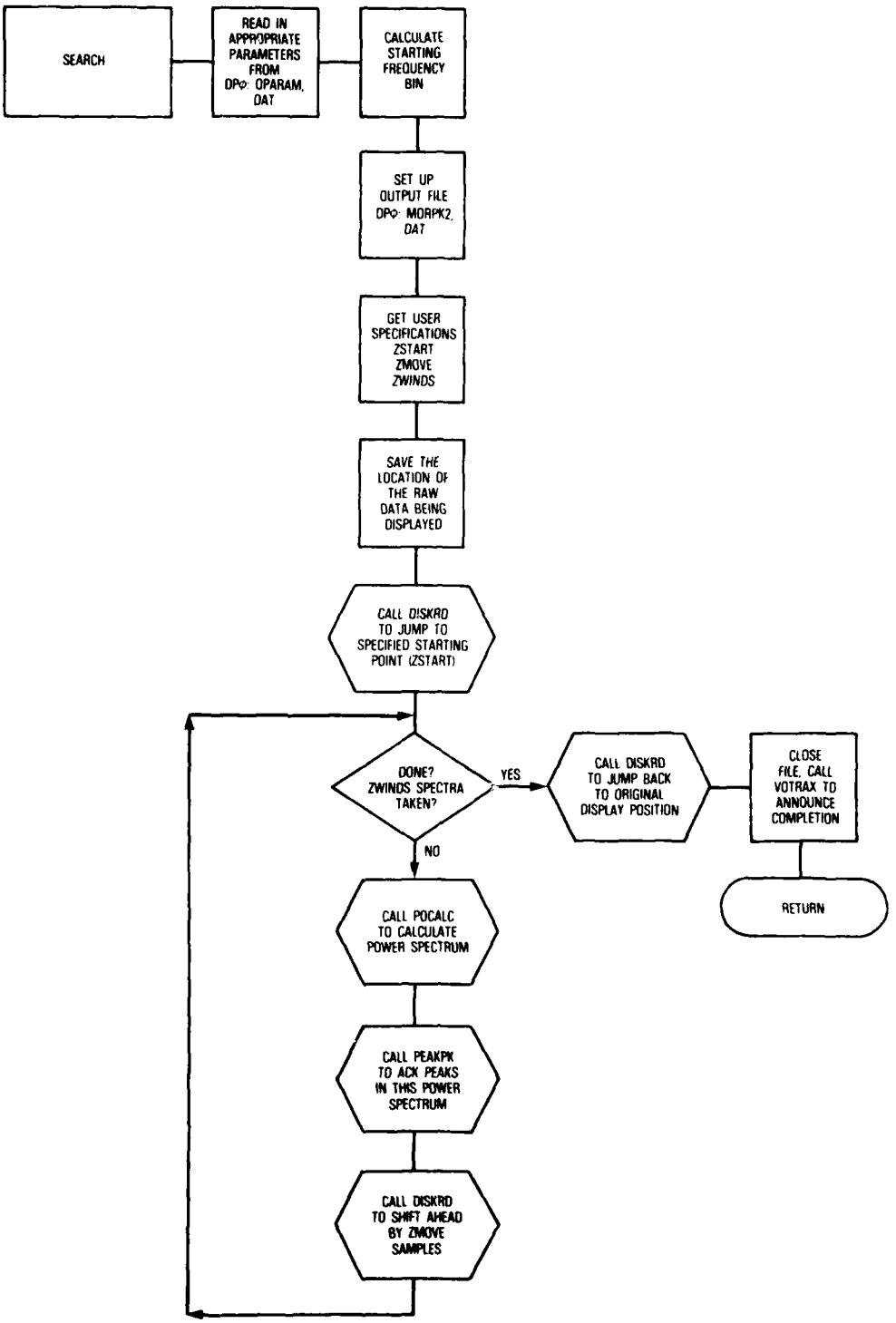


Figure 15 - Subroutine SEARCH Flowchart

SUBROUTINE NAME SHOUT (WHAT)

PURPOSE To issue voiced error responses by calling VOTRAX through VOCAL2.

DESCRIPTION

INPUT

OUTPUT

CALLED BY	CHINFO	CALLS	VOCAL2	GRAPHICS?	No
	SEARCH			DISK I/O?	No

SUBROUTINES CALLED

CALLING SEQUENCE

SUBROUTINE NAME STRA10 (ZSPEC, POWPTS)

PURPOSE To straighten phases of power spectra.

DESCRIPTION

INPUT

OUTPUT

CALLED BY	POCALC	CALLS	None	GRAPHICS?	No
SUBROUTINES CALLED				DISK I/O?	No

CALLING SEQUENCE

SUBROUTINE NAME TAPMOV (NCHAN)

PURPOSE To call the DR11C programs to transfer data from tape to disk.

DESCRIPTION

INPUT

OUTPUT

CALLED BY	VU	CALLS	DR11C1, DR11C2,	GRAPHICS?	No
SUBROUTINES CALLED			DREND	DISK I/O?	Yes
CALLING SEQUENCE				[Write to user-specified file]	

SUBROUTINE NAME TYPEIT (NUM1, ZNUM1, TRKNUM)

PURPOSE To type out ZNUM1 values of the raw data.

DESCRIPTION

INPUT

OUTPUT

CALLED BY KLOOK
VU

CALLS None

GRAPHICS? No

DISK I/O? No

SUBROUTINES CALLED

CALLING SEQUENCE

SUBROUTINE NAME UPDAT

PURPOSE To update the "Q" and "P" parameters in KLASIT. These parameters tell what the values of internal parameters were on previous forms.

DESCRIPTION This routine saves the internal variables of KLASIT for use if the most recent forms must be eliminated because of a narrow peak. (If the routine skips back several forms, it is desirable to know what the prevailing parameters were.)

UPDAT sets the "P" parameters to the "Q" parameters (thus saving the "Q" parameters), then sets the "Q" parameters to the "A" parameters. When the first sample is being used, the "Q" parameters must be initialized.

The flowchart for UPDAT is shown in Figure 16.

INPUT (Calling Sequence)

(DISK)

OUTPUT (Calling Sequence)

(DISK)

Commons: PRAM1 original parameters
PRAM2 "P" parameters
PRAM3 "Q" parameters
PRAM5 "A" parameters

CALLED BY KLASIT, LEV, NAROPK, NONLEV

CALLS None

SUBROUTINES CALLED

CALLING SEQUENCE CALL UPDAT

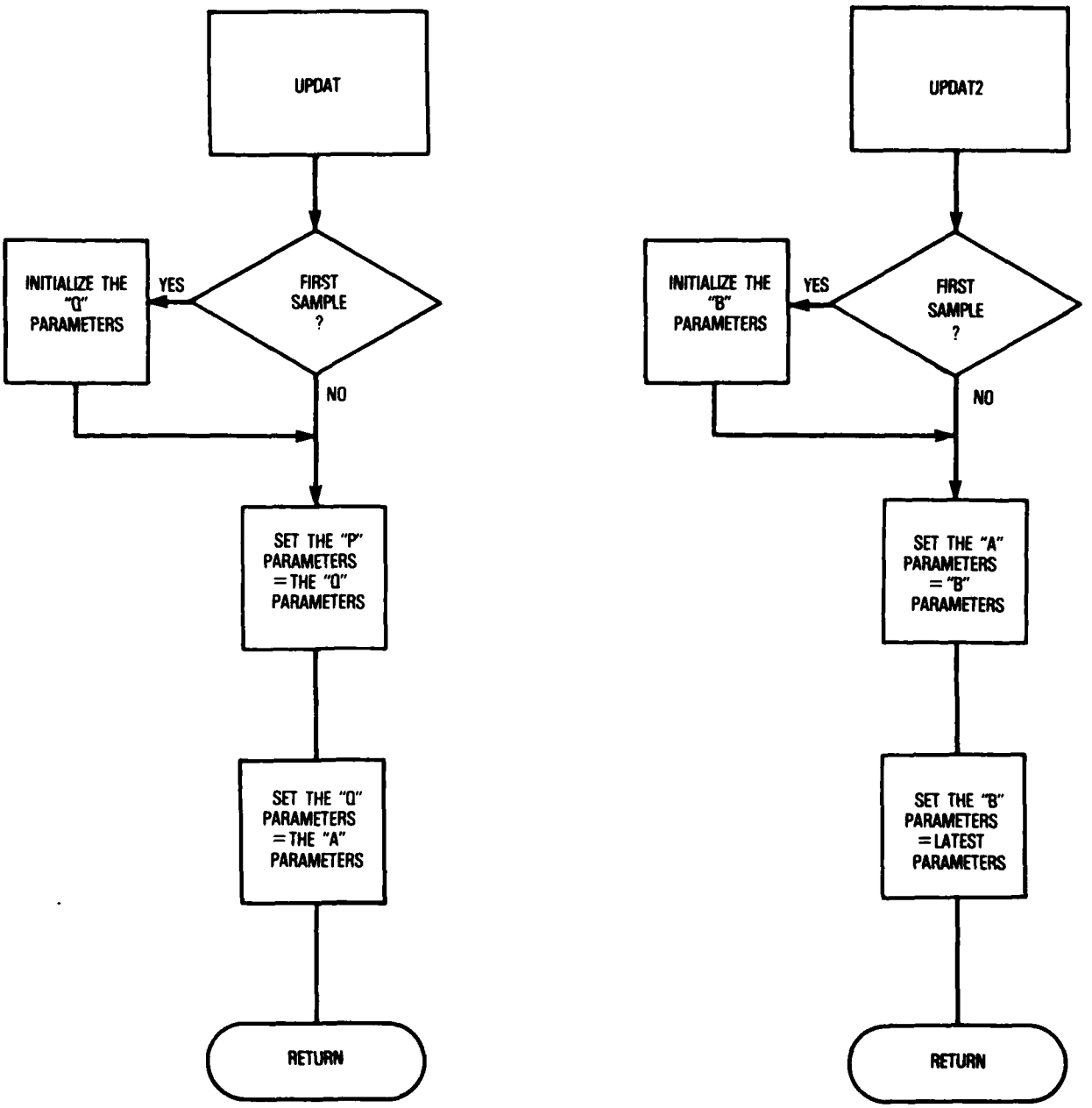


Figure 16 - Subroutines UPDAT and UPDAT2 Flowcharts

SUBROUTINE NAME UPDAT2

PURPOSE To update the "A" and "B" parameters in KLASIT. These parameters tell what the values of KLASIT internal parameters were on previous forms.

DESCRIPTION This routine saves the internal variables of KLASIT for use if the most recent forms must be eliminated because of a narrow peak. (If the routine skips back several forms, it is desirable to know what the prevailing parameters were.) The flowchart for UPDAT2 is shown in Figure 17.

UPDAT2 sets the "A" parameters to the "B" parameters (thus saving the "B" parameters) and then sets the "B" parameters to the current parameters. (The "B" parameters are initialized to the current parameters if L=1.)

The relative ages of these sets of parameters are shown in Figure 16.

INPUT (Calling Sequence)

OUTPUT (Calling Sequence)

(DISK)

Commons:	PRAM1	original parameters
	PRAM5	"A" parameters
	PRAM6	"B" parameters

CALLED BY KLASIT, LEV CALLS None

SUBROUTINES CALLED

CALLING SEQUENCE CALL UPDAT2

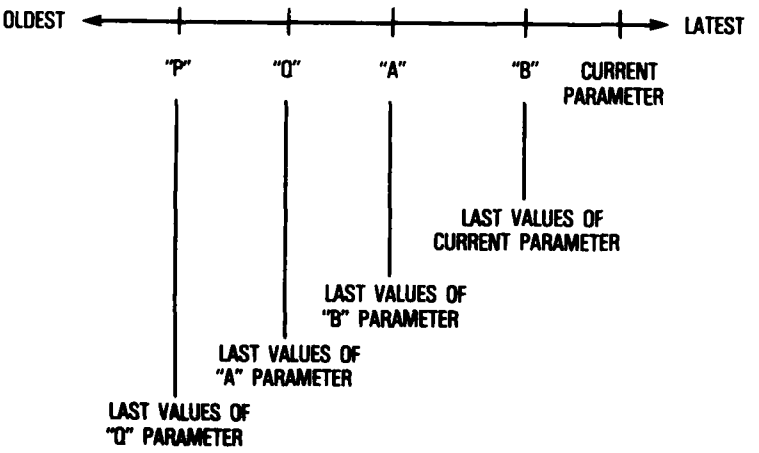


Figure 17 - Relative Ages of UPDAT and UPDAT2 Parameters

SUBROUTINE NAME VU (MAIN PROGRAM)

PURPOSE To provide excecutive program control for the graphics display program.

DESCRIPTION

OUTPUT

CALLED BY None

CALLS CHINFO, DISKRD, PICTS, DSPRAW, FFTDSP, POWDSP KLOOK, SEARCH, ADJUST, TAPMOV, SAVEIT, TYPEIT, XYPLOT, LOCATE, QVUCSI

GRAPHICS: Yes

DISK I/O: No

SUBROUTINES CALLED

CALLING SEQUENCE

SUBROUTINE NAME WAMDSP

PURPOSE To display WAMSER results.

DESCRIPTION WAMDSP reads the WAMSER approximation to the original data from the disk file DPO:WAMSER.DAT. WAMDSP reads each waveform triplet and calls NEWVEC to generate the corresponding display in a manner similar to subroutine DSPIT3. WAMDSP uses the decay time element of the WAMSER triplet to break up rises and falls into two line segments joined at x equal to the last x displayed plus decay time (the second segment running from last x + decay time to last x plus duration). The display is turned on and given the tag and status variable MWAMS. Before returning, WAMDSP also generates a copy of the MWAMS display both at the top of the screen and at the bottom (tagged TWAMS and BWAMS) and leaves the top one on if "W1" was the last user command and leaves the bottom one on if "W2" was the command.

INPUT Disk: DPO:WAMSER.DAT

A FORTRAN direct-access file with 1024 records consisting of three words each. Record 1024 contains the number of actual records composing the approximation. Each triplet consists of a level value, decay time, and form duration.

Common

/CMNDS/NUM3 - 2nd character of last user command

OUTPUT

Common

/DCHEK/MWAMS, TWAMS, and BWAMS - status variables of the three WAMSER displays

CALLED BY KLOOK

SUBROUTINES CALLED NEWVEC
(graphics library)

CALLING SEQUENCE CALL WAMDSP

SUBROUTINE WAMSER

PURPOSE To smooth transitions between level forms and rise or fall forms in KLASIT output.

DESCRIPTION Subroutine WAMSER performs final smoothing on the waveform data produced by subroutine KLASIT and stored on disk file DPO:FINAL.DAT. WAMSER refines the boundary points of rise and fall transitions within the KLASIT data. In the case of consecutive non-level forms, the start of the second transition (and end of the first transition) is the point at which direction changes as provided by KLASIT. Should a rise or fall be preceded by a level, the transition begins at the end of the first series of points in the original data of minimum length, RUN, in the level which contains only "isolated" jumps, if any, in the direction of the transition. An isolated jump is a change occurring at a point whose two successors are at the same level as its predecessor. In a similar manner, if a level follows a rise or fall, the transition associated with the rise or fall ends in the level region at the endpoint, working backwards, of a similar "run" of points in the level. If the minimum run length (RUN) is not exceeded, then the transition begins at the point at which the change in amplitude from the level amplitude exceeds a percentage (parameter BAND) of the total change represented by the rise or fall.

After transition boundary points are determined, the new durations of the forms are calculated and a decay time is added to the form triplet to provide a better representation of the curve shape. The decay time is the number of points required for the transition to achieve a percentage (parameter TEN) of the total change in level. The type of form, duration, level, and decay times are stored in triplets and written to the disk file DPO:WAMSER.DAT as described in the OUTPUT paragraph.

KLASIT triplets are read from disk one at a time, and the new WAMSER triplets are stored on the output disk file as they are created. The routine retains copies of the previous two WAMSER forms and the next KLASIT form surrounding the current form triplet.

The WAMSER flowchart is shown in Figure 18.

NAME WAMSER (Cont'd)

INPUT

Arguments

ARRAY - original data array - raw data

LENGTH - length of ARRAY

TEN - parameter (real valued) containing percentage factor for decay time computation

BAND - floating point parameter containing level change threshold for alternate method of marking transition start or stop.

DISK

DPO:FINAL.DAT - file containing triplets comprising KLASIT approximation to original data contained in ARRAY

Data Statement

RUN = 4 - parameter containing minimum number of points for run

OUTPUT

DISK

DPO:WAMSER.DAT - file containing triplet comprising WAMSER approximation to original data
(1024 records, three words per record, FORTRAN direct access. Record number 1024 contains NROWS - the total number of triplets in WAMSER smoothed signal. Each triplet defined as follows:

KCOL1 = value of level

KCOL2 = 0 if this form is a level, or
= decay time at start of the rise or fall

KCOL3 = duration of the form, or
= 0, if this is end level)

CALLED BY KLOOK

SUBROUTINES CALLED None

CALLING SEQUENCE CALL WAMSER (ARRAY, LENGTH, NROWS, TEN, BAND)

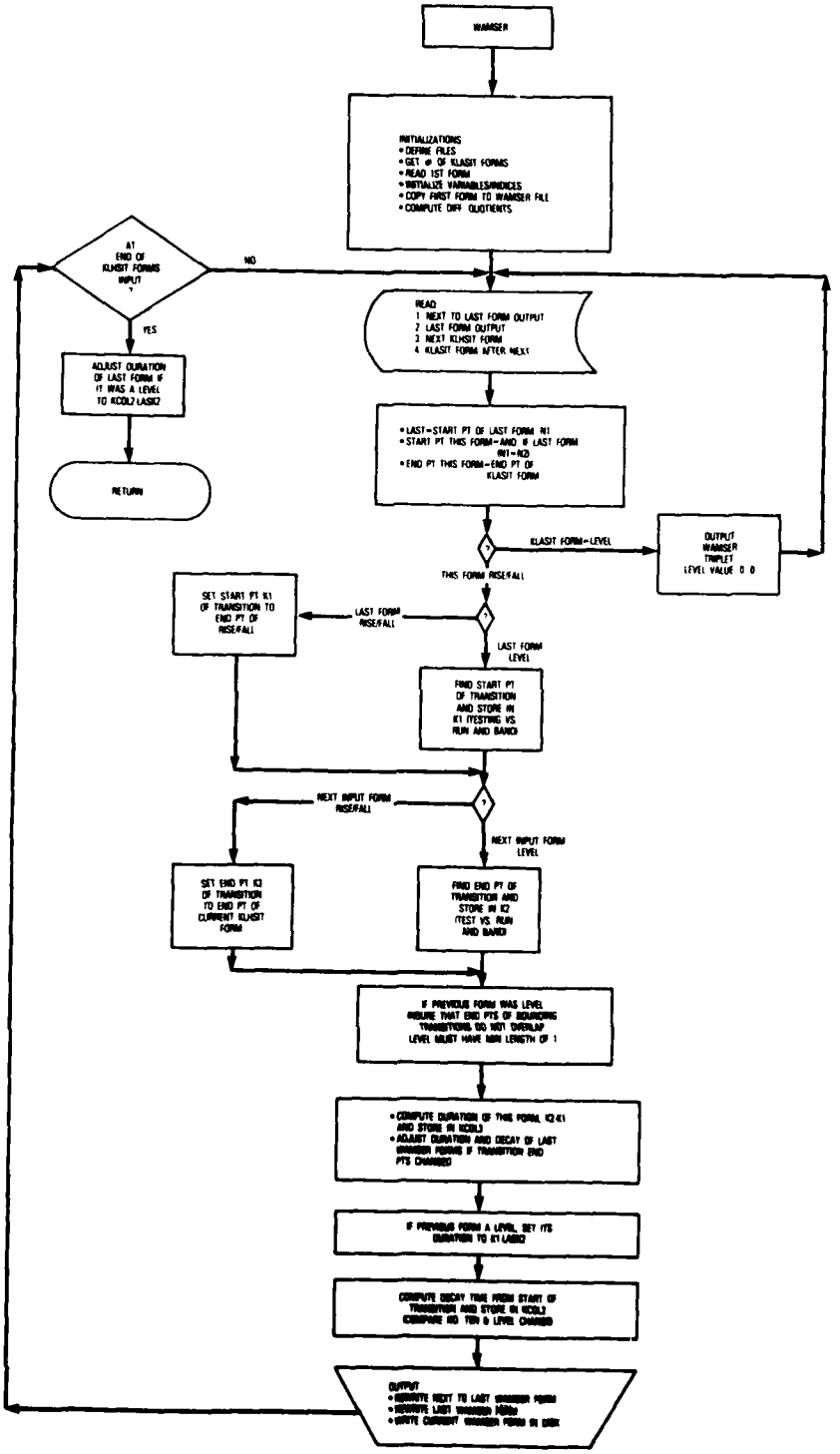


Figure 18 - Subroutine WAMSER Flowchart

SUBROUTINE NAME VOCAL2

PURPOSE To allow sentences to be enunciated by VOTRAX.

DESCRIPTION

INPUT

OUTPUT

CALLED BY SHOUT | CALLS None | GRAPHICS: No

SUBROUTINES CALLED | DISK I/O: No

CALLING SEQUENCE

SUBROUTINE NAME XYPLOT (BUFFER)

PURPOSE To produce a hardcopy of the current display.

DESCRIPTION

INPUT

OUTPUT

CALLED BY SELECT | CALLS None | GRAPHICS: No

KLOOK | DISK I/O: No

VU

SUBROUTINES CALLED

CALLING SEQUENCE

VU FILE DESCRIPTIONS

Input Data File

The input file is a non-FORTRAN, unformatted disk file. Each block on the file contains 256 words (16 bits), each is an integer value. The blocks are read one at a time as the data in the block are required. A specified block is read using a FORTRAN callable RT-11 SYSLIB function IREADW. The statement is

```
WRDSRD=IREADW (256,DSKBUF,IBLK, ICHAN)
```

which reads 256 words from a block IBLK on a file associated with ICHAN. The words are stored in a buffer called DSKBUF. The acutal number of words read is stored in WRDSRD. All variables are integer values.

Parameters File

The user adjustable parameters are stored on a file named DPO:OPARAM.DAT. This file contains both the current parameters and the default parameters. This file is created with two FORTRAN write statements as follows:

```
WRITE(98) AMPMIN,SFREQ,LOGPOW,AMPBEG, (IDEF(I),I-1,4)
WRITE(98) BAND,PERHI,PERLO,FACTOR,FMERGE,HMERGE,BWIDTH,ZPSCAL,
1         (ZDEF(I),I-1,8)
```

AMPMIN, SFREQ, LOGPOW, AMPBEG, and IDEF are INTEGER*2 variables. BAND, PERHI, PERLO, FACTOR, FMERGE, HMERGE, BWIDTH, ZPSCAL, and ZDEF are REAL*4 variables.

Window Parameters

The Kaiser-Bessel window parameters are generated and stored on a file named DPO:WINDOW.DAT. This file is created with FORTRAN unformatted direct access I/O. The file, assigned as logical unit 71, is defined with a length of 1025 records where each record contains two words. The FORTRAN definition statement is

```
DEFINE FILE 71 (1025,2,U,ASS71)
```

The windowing parameters are stored by creating PTSDPY records. The FORTRAN statement is

```
WRITE(71'I) ZWINDO
```

where I runs from I to PTSDPY, and ZWINDO is the real Kaiser-Bessel window parameter. The 1025 record contains PTSDPY and is written with FORTRAN statement

```
WRITE(71'1025) PTSDPY
```

PTSDPY is an integer variable.

Peakpick Results

The peakpicking results generated for SELECT (the "Z" command) are stored on a file named DPO:MORPK2.DAT. This file is created with FORTRAN unformatted direct access I/O. The file, assigned to logical unit 77, is defined with a length of 1024 records where each record is one more than the maximum number of peaks (currently 10). The FORTRAN definition statement is

```
DEFINE FILE 77 (1024,RSIZE,U,ASS77)
```

where RSIZE is an integer equal to the number of peaks plus 1. The last record (1024th) is first written with FORTRAN statement

```
WRITE(77'1024) ZSTART,ZMOVE,NWINDS,PTSDPY
```

ZSTART and ZMOVE are REAL*4 variables representing the first sample and the shift duration, respectively. NWINDS and PTSDPY are INTEGER*2 variables representing the number of windows and the points displayed, respectively. The peaks are written on disk, one window per record, starting with record one. The FORTRAN statement is

```
WRITE(77'JWNDOO LTYPE,(PPOUT(I),I=1,NOP)
```

where JWNDOW runs from 1 to NWINDS, and NOP is the number of peaks. LTYPE and PPOUT are INTEGER*2 variables. PPOUT contains peakpicking results.

Display Buffer Save Files

As many as five display buffers may be saved on five files with the "T5" function. The buffers are stored on consecutive files named DPO:TESTOO.DPY, DPO:TEST01.DPY, DPO:TEST02.DPY, DPO:TEST03.DPY and DPO:TEST04.DPY. Each buffer is saved using the FORTRAN statement

```
CALL SAVE(DPO:TESTOO)
```

which is described in the FORTRAN extensions for the VT11 Graphics Support. The file name is updated for each consecutive call.

KLASIT Output Files

There are three KLASIT output files and all have the same format. They are all unformatted FORTRAN direct access files with 1024 three-word records. The last record contains only the actual number of data records on the file.

The three files are

The First Approximation to the Unsmoothed Data (UNSMOO)

The Smoothed Noise (SMOO)

The Final (KLASIT) Smoothed Approximation to the Original Data (BFIN2)

The files are set up by the following assignments:

```
CALL ASSIGN (79, 'DPO:UNSMOO.DAT', 14) or
CALL ASSIGN (79, 'DPO:SMOO.DAT', 12)
CALL ASSIGN (79, 'DPO:FINAL.DAT', 13) and
DEFINE FILE 79 (1024,3,U,ASS79)
```

Each three-word record consist of

1. 0, 1, or -1 indicator for level, rise, or fall
2. endpoint of the form
3. value of the level

WAMSER Output File

There is one WAMSER output file containing the WAMSER smoothed approximation to the original data. It is an unformatted FORTRAN direct access file containing 1024 three-word records. Each record corresponds to and describes a linear form used to approximate the original data in an indicated time interval. The last record on file contains the count of the forms used to form the approximation. The file is set up by the following statements:

```
CALL ASSIGN (80, 'DPO:WAMSER.DAT', 14)
DEFINE FILE 80 (1024, 3, u, ASS80)
```

Each three-word record consists of

1. value of level (or value of rise/fall at end)
2. 0 if form is a level or, if not a level, decay time
3. duration of form (or zero if this is the end level)

KLASIT Scratch File

The input array containing data to be smoothed by KLASIT is written out to a scratch file so that it is available for recovery purposes. The file is a FORTRAN unformatted binary file created by the statements:

```
CALL ASSIGN (81, 'DPO:KLSINP.DAT', 14)
WRITE (81) LENGTH, (S(I),I-1,LENGTH)
```

VU LINKING PROCEDURE

Figure 19 shows the RT-11 BATCH file required to generate the VU execution module. The subroutine object files are on DPO. These object files are created by compiling the FORTRAN routines and by assembling the MACRO routines. VTLIB is the RT-11 library containing the graphics display drivers. SYSLIB is the FORTRAN callable system subroutine library. The FORTRAN object time library is attached with the /F switch in the first link statement. The /C switches are command line continuation indicators.

```
$JOB
$RUN LINK
$DATA
DP0:VUE500],DP0:VU=DP0:VU,SY:VTLIB,LPSLIB,SYSLIB/B:2500/F/C
DP0:KCNTRL/O:1/C
DP0:LOCATE/O:1/C
DP0:SAVEIT/O:1/C
DP0:FOWDSP/O:2/C
DP0:FFTDSP/O:2/C
DP0:TAPMOV/O:2/C
DP0:ADJUST/O:2/C
DP0:DSPRAW/O:2/C
DP0:DISPLS/O:2/C
DP0:WAMDSP/O:2/C
DP0:KCREAT/O:2/C
DP0:KCMNDS/O:2/C
DP0:CHANGE/O:2/C
DP0:DTRMIN,PDATA/O:2/C
DP0:KLASIT/O:2/C
DP0:KLOUT/O:2/C
DP0:WAMSER/O:2/C
DP0:SEARCH/O:2/C
DP0:XYPLOT,XYHDLR/O:2/C
DP0:VUCST1/O:3/C
DP0:CHINFO/O:3/C
DP0:FOCALC/O:3/C
DP0:FOWSUB/O:3/C
DP0:FFKSUB/O:3/C
DP0:ADJSUB/O:3/C
DP0:PRAMTR/O:3/C
DP0:TYPEIT/O:3/C
DP0:PEAKFN/O:3/C
DP0:KNOLEV/O:3/C
DP0:KLEVEL/O:3/C
DP0:DISKRD/O:3/C
DP0:MAGTAP/O:3/C
DP0:FOURIA/O:4/C
DP0:KNARDP,KUPDT1/O:4/C
DP0:KINSRT/O:4/C
DP0:NEWVEC,PICTS/O:4/C
DP0:SHOUT,VOCAL2/O:4/C
DP0:KUPDT2/O:4/C
DP0:STRA10/O:4/C
DP0:MRGEFL/O:4/C
DP0:PATFN/O:4/C
DP0:REEDTR/O:4
$EOD
$EOJ
```

Figure 19 - BATCH File to Link VU

PROGRAM SELECT

SELECT is an interactive graphics program for the examination, selection, and tracking of peaks in power spectra. It is coded in FORTRAN and runs in the same PDP 11/45 minicomputer system as does program VU. SELECT operates on data supplied to it by program VU in the form of frequency bin numbers associated with peaks in individual power spectra. Each list of numbers associated with a single power spectrum is called a "spectral slice." SELECT allows the user to examine an ensemble of spectral slice data and track peaks throughout the ensemble. The functional overview of SELECT as well as its command language are contained in the ISPARS User's Guide.[1] This section presents, alphabetically by subroutine name, the subroutine descriptions, input/output, files, and linking procedures for program SELECT.

SELECT SUBROUTINE DESCRIPTIONS

SELECT

PURPOSE To mark and store selected spectral peak data, perform tracking of spectral peaks through this data using selected parameter values, and store tracking results on disk.

DESCRIPTION SELECT is the central control program for analysis and display of peaked spectral data. SELECT accesses this spectral data from the file DPO:MORPK2.DAT created by program VU and displays the first thirty slices of data. SELECT then awaits user commands to continue processing the data. The user may examine the entire file of peaked spectral slices using "UP" and "DOWN" commands and can select and mark any number of slices for future processing by the tracking procedure. User slice selections are indicated on the current display by changing the baseline for the slice from a solid to a dashed line. When the selections are complete, the designated spectral slices are retained on the disk file, DPO:TRKABL.DAT, which serves as input to the tracking procedures when they are initiated by the appropriate user command. The results of tracking are stored on disk file, DPO:TRAKED.DAT, and can be displayed on user request. The two files produced by SELECT, DPO:TRKABL.DAT and DPO: TRAKED.DAT, as well as the input file, DPO:MORPK2.DAT, are retained on disk after completion of the program so that, in subsequent program executions, the selection and tracking procedures can be skipped and the tracking results from the last execution of SELECT can be displayed immediately upon request from the user.

DPO:TPARAM.DAT is another input and output file to SELECT and contains both the default values of parameters used in tracking or displaying the data and also the last used values of these parameters. The user, in the course of processing the peaked spectral data, may change parameter values and generate new tracked data using these new parameters. The parameter values used by the program are saved to be available for use the next time the program is executed if the user so desires. The default values are always saved to allow the user to return to them whenever he wishes; however, the default values themselves may also be changed by user request and the new default values will be retained on file.

The first operation performed by SELECT is the display of the first thirty slices of peaked spectral data. The program then waits for user instructions

via keyboard commands of the form given in the table which follows. Illegal commands are ignored and the program waits for a legal instruction. All commands begin with an alphabetic character. Some commands may consist of a letter and one or two two-digit numbers, separated by a space as specified in the table.

TABLE D1 - SELECT COMMANDS

Command	Action	Subroutines Called
E	STOP; exit program	
An	Mark or unmark the spectral slice numbered n (max. 99)	MARKIT
M	Move marked slices from DPI:MORPK2.DAT to the file DPO.TRKABL.DAT and display the latter	MOVE2 XPKDSP
M1	Display the current contents of DPO:TRKABL.DAT	XPKDSP
T	Perform tracking on contents of DPO:TRKABL.DAT and store results on DPO:TRAKED.DAT	XSTR8S XSETF3
T1	Display tracking results or erase them if already displayed	XTDISP
T2	Erase everything else and display only tracker results, or erase them if they are already displayed	XTDISP
G	Produce hard-copy of current display	XYPLOT
R	Reinitialize display buffer and reenter program logic by displaying the beginning of DOP:MORPK2.DAT	XPKDSP
Unn Dnn	Move current display up/down by nn lines where nn is 2-digit entry	XUP XPKDSP XTDISP
B	Re-display original time slices from DOP:MORPK2.DAT	XPKDSP
Cnn mm	(refers only to MORPK2 data) Mark or unmark the spectral slices numbered nn through mm	BUNCH XPKDSP
C3	Mark (or unmark) all spectral slices in MORPK2.DAT	BUNCH XPKDSP
P	Use potentiometer to determine frequencies	POTTY

TABLE D1 - SELECT COMMANDS (Cont'd)

Command	Action	Subroutines Called
W	Print spectral slice location in original data	WHERE
"BLANK" or <CR>*	Move display up or down according to default preset values	XBLANK
X	Display parameters with options to change them and then redisplay tracks or spectral slices whichever were last displayed	XADJST

*carriage return

INPUT keyboard input - FORMAT (1A1, 12, 1X, 12)

CMAND1 - command letter

NUM1 - first two-digit numeral of command, if any

NUM2 - second two-digit numeral of command, if any

OUTPUT None

SUBROUTINES CALLED XPKDSP

MARKIT

XSTR8S

XSETF3

XTDISP

XYPLOT

XUP

BUNCH

POTTY

XBLANK

WHERE

XADJST

CALLED BY None

SELECT is a program.

CALLING SEQUENCE None

SUBROUTINE NAME XTDISP

PURPOSE To display tracks

DESCRIPTION Subroutine XTDISP displays the contents of the file, DPO:TRAKED.DAT, within the spectral slice number and frequency bounds set up by the user in other parts of program SELECT. This file contains all the tracks found in the spectral data by the tracking routines. The track display covers 30 spectral slices along the y-axis, increasing in index from the top of the screen to the bottom. The starting spectral index of the display is specified by BEGIN2, the argument to subroutine XTDISP. The frequency bounds of the display are specified by the starting frequency ISTART, specified earlier by the user and stored on file DPO:TPARAM.DAT, and by the window size PTSDPY of the FFT's used to generate spectral data. PTSDPY is contained in the 101st record of DPO:TRAKED.DAT. The maximum frequency possible is 9 KHz since the sample rate of the raw acoustic data was 18 KHz. The internal size of each power spectrum frequency bin is, therefore, 9000 (PTSDPY 2-1). The lower frequency bound of the display is that bin containing the requested start frequency, and the range of frequencies is 51 bins.

Subroutine XTDISP examines all tracks stored on file and displays all parts of all tracks which intersect the bounds of the display. Tracks are displayed by vectors between points where the slope of the track changes. The track display is a subpicture with tag equal to 50.

INPUT

Arguments

BEGIN2 - spectral slice index for start of display

Files

DPO:TPARAM.DAT - ISTART (word #7 of record 1) lower frequency bound of display

DPO:TRAKED.DAT - (unit 11) (FORTRAN direct access)

record #101

TRKTOT - total number of tracks

TOT99 - total number of spectral slices

PTSDPY - window size of FFT

record #n

track number n - list of frequency bin numbers read into THSTRK array

SUBROUTINE NAME XTDISP (Cont'd)

OUTPUT Track display

SUBROUTINES CALLED None

CALLED BY SELECT

CALLING SEQUENCE CALL XTDISP (BEGIN2)

SUBROUTINE NAME XADJST

PURPOSE To display current or default program parameters and allow user to change parameters via specified keyboard commands.

DESCRIPTION Seven parameters are used within program SELECT and retained on disk file DPO:TPARAM.DAT. The first seven words of this file contain the current "working" values of these parameters and the next and final seven words contain default values for these same parameters. Subroutine XADJST reads the values for the list of program parameters. The routine is called by SELECT when prompted by the curser commands "X" or X1". If the second character of the command is "1", XADJST sets the program parameters to the default values retained on TPARAM.DAT. Otherwise, the parameters are set to the "working" values currently on file. These may be values saved earlier in the current execution of SELECT or in a preceding execution.

After the initial data input and assignments, the program variables and current values are displayed, accompanied by a notation of the command necessary to change each one.

XADJST then waits for a command from the keyboard; possible commands and actions taken are:

F1 = N - set FROGAP equal to N

K1 = N - change KNTLIM to N

T1 = N - change TRACKS to N

T2 = N - change TIMGAP to N

I1 = N - change ISTART to N

N1 = N - change NUM1 to N

C - changes CMAND1. (If already equal to U, sets it to D and vice versa)

X - erase display; save changes in working values only and output all parameter values to file TPARAM.DAT and return

X1 - erase display; reset defaults to new working values; output all values to TPARAM.DAT and return

E - exit program

E1 - do same as X and X1, respectively, but exit program instead of return.

SUBROUTINE NAME XADJST (continued)

INPUT

Arguments

NUMX - argument containing the second character entered by the user when initiating the call to XADJST the number "1" is the only meaningful value and causes parameters to be set to default values.

File

DPO:TPARAM.DAT - (unit 98) (unformatted binary)
One record containing seven working parameter values and seven default parameter values

OUTPUT

File

DPO:TPARAM.DAT - (cf above) on output, contains updated values

SUBROUTINES CALLED None

CALLED BY SELECT

CALLING SEQUENCE CALL XADJST (NUMX)

SUBROUTINE NAME XUP

PURPOSE To compute the index of the spectral slice which should appear at the top of the display when a shift up or shift down command has been received.

DESCRIPTION The given shift number is added to or subtracted from (for shift up or shift down, respectively) the index of the current slice at the top of the display. If the shift number is zero, shift to the first spectral slice for down shift and the spectral slice that is thirty from the last in the case of up shift. Whenever the top index is within thirty of the end of data, it is reset to the slice thirty from the end.

This routine handles both displays of the original spectral slices and also selected slices and tracking results. The argument PNUM serves as a means of indicating which type of data is referred to by the shift command.

INPUT

Arguments

PNUM - if equal to one, commands refer to original spectral slices; otherwise, they refer to selected slices

BEGIN - index of spectral slice currently at top of display of original spectral slices

BEGIN2 - index of spectral slice currently at top of display of selected slices

CMAND1 - Hollerith variable containing "U" or "D" to indicate direction of shift

NUM1 - integer amount of shift

COMMON [CHUNKS] - contains NWINDS, the total number of spectral slices

OUTPUT

Arguments

BEGIN and BEGIN2 - new values

SUBROUTINES CALLED None

CALLED BY SELECT

CALLING SEQUENCE CALL XUP (PNUM,BEGIN,BEGIN2,CMAND1,NUM1)

SUBROUTINE NAME POTTY

PURPOSE To display a vertical cursor on screen controlled by the potentiometer for channel zero and to display in the console LED's the frequency value of the cursor position on the frequency axis.

DESCRIPTION A vertical cursor, whose position is movable and depends on the value of the A/D conversion of the signal in channel zero, is displayed. With no signal physically plugged into the console in this channel, the signal sampled is the potentimeter output which varies from 0 to 4096 digitally. An 18000-Hz sample rate is assumed for the spectral slice input data, and hence frequency intervals on the horizontal axis equal 9000/PTSDPY/2-1), where PTSDPY is the number of points taken in the FFT. The position of the cursor is converted to frequency and the frequency is displayed in the console LED's.

INPUT

Arguments

PTSDPY - number of points taken in FFT

File

DPO:TPARAM.DAT - (unit 98) (FORTRAN unformatted binary)
seventh word on 1st record is start frequency for display

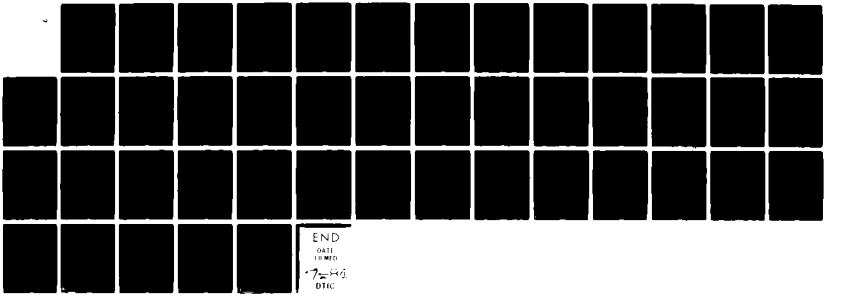
OUTPUT frequency value of cursor position displayed in console LED's

SUBROUTINES CALLED None

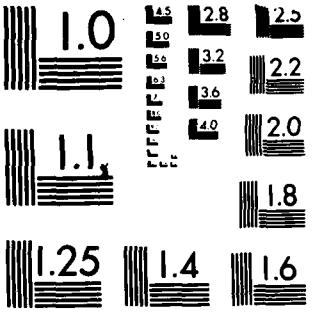
CALLED BY SELECT

CALLING SEQUENCE CALL POTTY (PTSDPY)

AD-A141 575 PROGRAM DESCRIPTIONS FOR INTERACTIVE SIGNAL AND PATTERN 2/2
ANALYSIS AND RECO..(U) DAVID W TAYLOR NAVAL SHIP
RESEARCH AND DEVELOPMENT CENTER BET.. W PARSONS ET AL.
UNCLASSIFIED MAR 84 DTNSRDC/CMLD-84-05 F/G 9/2 NL



END
DATE FILMED
7-84
DTIC



SUBROUTINE NAME WHERE

PURPOSE For an indicated series of spectral slices in DPO:TRKABL.DAT, to print out the absolute location of each in original peaked data MORPK2.DAT.

DESCRIPTION Subroutine WHERE begins at a specified record (spectral slice) in DPO:TRKABL.DAT and reads a sequence of records corresponding to spectral slices from TRKABL.DAT in order to print out the second word of each record which contains the index of that record in the original data.

INPUT

Arguments

BEGIN2 - TRKABL.DAT index of first slice of display

NUM1 - index of first slice for which information is desired. This is relative to display. Must be between 1 and 30.

NUM2 - index of last slice for which information is desired. Must be between 0 and 30. If equal to zero, only NUM1th record is printed.

File

DPO:TRKABL.DAT

OUTPUT

TTY - index relative to display and absolute index in DPO:MORPK2.DAT for each spectral slice indicated by input

SUBROUTINE CALLED None

CALLED BY SELECT

CALLING SEQUENCE CALL WHERE (BEGIN2, NUM1, NUM2)

SUBROUTINE NAME XBLANK

PURPOSE To access fifth and sixth words on DPO:TPARAM.DAT and return values through arguments.

DISCRIPTION Performs: READ(98) TRACKS, GROGAP,KNTLIM,TIMGAP,CMAND1,NUM1 and returns CMAND1 and NUM1 to calling routine. These are current working values for the direction and amount of the automatic move commands.

INPUT

File

DPO:TPARAM.DAT - (unit 98)(FORTRAN unformatted binary) (Single record, cf SELECT File Descriptions)

OUTPUT

Arguments

CMAND1 - from disk

NUM1 - from disk

SUBROUTINES CALLED None

CALLED BY SELECT

CALLING SEQUENCE CALL XBLANK (CMAND1,NUM1)

SUBROUTINE NAME BUNCH

PURPOSE To change the display line type of a series of consecutive spectral slices on DPO:MORPK2.DAT.

DESCRIPTION The first two arguments to BUNCH are the beginning and end record numbers identifying a series of spectral slices on DPO:MORPK2.DAT. BUNCH reads each of these records and changes the type of the base line associated with each from solid to broken or broken to solid, depending on what it is initially. The records with the new line types are replaced on DPO:MORPK2.DAT. No display is changed.

INPUT

Arguments

NUM1 - begin and end absolute record numbers

NUM2 - of slices on MORPK2 to be changed

BEGIN - not used or altered

COMMON/CHUNKS/ cf MOVE 2

File

DPO:MORPK2.DAT

OUTPUT

DPO:MORPK2.DAT

SUBROUTINES CALLED None

CALLED BY SELECT

CALLING SEQUENCE CALL BUNCH (NUM1,NUM2,BEGIN)

SUBROUTINE NAME MOVE2

PURPOSE To transfer spectral slice data from DPO:MORPK2.DAT to DPO:TRKABL.DAT for each slice chosen by an "A" command in the main routine.

DESCRIPTION Each record of DPO:MORPK2.DAT is read separately and the line type indicator, LTYPE, (the first word of the record) is checked. If LTYPE=2, the record is written out to the file DPO:TRKABL.DAT with the original record number on MORPK2 added after LTYPE. The records written in DPO:TRKABL.DAT are counted and the total is added to the list of data obtained from record number 1024 of MORPK2.DAT, to form record 1024 of TRKABL.DAT.

INPUT

COMMON /CHUNKS/ - contains NOP, the maximum number of peaks in each spectral slice. Determines the record size on DPO:MORPK2.DAT.

File

DPO:TRKABL.DAT - (unit 77)(FORTRAN direct access)
File containing peaked spectral slice data. Maximum number of records is 1024; record size is NOP+1.

OUTPUT

File

DPO:TRKABL.DAT - (unit 99)(FORTRAN direct access)
File containing only selected spectral slices. Record format same as MORPK2.DAT. except that record size is one larger and original record number of the slice on MORPK2 is added after LTYPE.

SUBROUTINES CALLED None

CALLED BY SELECT

CALLING SEQUENCE CALL MOVE2

SUBROUTINE NAME MARKIT

PURPOSE To change the base line of a designated spectral slice in the current display from solid to broken or vice versa.

DESCRIPTION MARKIT changes the manner of display for a particular spectral slice designated by the subroutine argument NUM1. NUM1 must refer to a spectral slice on display and must be a positive number no greater than thirty. The spectral slices on display are assumed to be from file DPO:MORPK2.DAT which forms one of the primary input files to program SELECT. The second argument to MARKIT, called BEGIN, identifies the first spectral slice of the display as to its record number in MORPK2.DAT.

MARKIT locates the designated slice in the file MORPK2.DAT, reads its associated data record, and changes the type indicator of the record from solid to broken or vice versa. This type indicator specifies whether the base line displayed for the spectral slice is solid or broken. When the type is changed, the altered data are replaced on the MORPK2.DAT file. The subpicture associated with the spectral slice is erased, recreated to blink for one second, erased again, and finally recreated and displayed with the new line type.

These operations are facilitated by creating each spectral slice as a separate subpicture and using its index on the data file MORPK2.DAT as the associated subpicture tag.

INPUT

Arguments

NUM1 - index of designated spectral slice (relative to top of display). (Must be no greater than 30).

BEGIN - record number of first spectral slice on display. Record number refers to its position in DPO:MORPK2.DAT.

COMMON /CHUNKS/ - contains NOP, the maximum number of peaks in each spectral slice

File

DPO:MORPK2DAT - (unit 77)(FORTRAN direct access) File containing data for one spectral slice per record.

Each record consists of a base line type to determine solid or broken line display (LTYPE=1 or 2) and a list of NOP numbers representing the frequency bin numbers of peaks in the associated power spectrum.

SUBROUTINE NAME MARKIT (Cont'd)

OUTPUT

File

DPO:MORPK2.DAT - same as on input except that the line type for the designated spectral slice is reversed

SUBROUTINE CALLED None

CALLED BY SELECT

CALLING SEQUENCE CALL MARKIT (NUM1,BEGIN)

SUBROUTINE NAME XPKDSP

PURPOSE To display contents of file containing spectral slice to be tracked or contents of file containing original output of peakpicking routine.

DESCRIPTION The arguments to subroutine XPKDSP specify the file of data to be displayed and the spectral slice number within that file at which to begin the display. The possible files are restricted to either the file of peaked spectral slices input to program SELECT or the file containing the selected spectral slices to be tracked. XPKDSP displays up to 30 spectral slices beginning with the specified record. Fifty-one frequency bins are displayed beginning at a user specified frequency obtained from disk file DPO:TPARAM.DAT. The presence of a peak is indicated by an "X" in the appropriate bin with frequency being the horizontal axis and slice number along tne vertical. The peak display for each spectral slice is a separate subpicture whose tag is the index of the slice in the current display.

INPUT

Arguments

BEGIN - spectral slice number at which to start display

DNUMBR - indicator of input file to be used

Common

/CHUNKS/,NOP - the number of peaks per spectral line

Files

DPO:TPARAM.DAT - (unit98) (unformatted binary) A single 7-word record. Word number seven, ISTART, contains frequency at which to begin display.

DPO:TRKABL.DAT - (unit 99) File containing the list of peaks (frequency bin numbers) for each spectral slice selected to be tracked, cf XSTR8S

DPO:MORPK2.DAT - (unit 99) File containing the list of peaks for each spectral slice output by the peakpicking procedures and input to SELECT

OUTPUT Output restricted to creation of new display buffer contents

SUBROUTINES CALLED Calls only FORTRAN library and system graphics routines

CALLED BY SELECT

CALLING SEQUENCE CALL XPKDSP(BEGIN,DNUMBR)

SUBROUTINE NAME XSTR8S

PURPOSE For each frequency in the input power spectra, to determine the spectral slices in which that frequency appears as a peak and the number of consecutive slices in which it remains a peak.

DESCRIPTION Input for this routine consists of the file, DPO:TRKABL.DAT, which contains the spectral slices of peaked data selected by the user in routine MARKIT. Each record consists of a list of bin numbers of the power spectrum peaks determined by the VU routine PEAKPK. The maximum number of peaks per power spectrum is contained in a common variable NOP currently set to ten.

XSTR8S searches each list of spectral peaks for the occurrence of each possible frequency bin (max=250). The frequency bin number, the start index in the spectral slices of its occurence as a peak, and the number of successive slices containing the peak are output as a three-word record on the FORTRAN direct access file DPO:SLINES.DAT. The maximum number of such records is 1023. The 1024th and last record is used to output the actual number of three-word records put on file. All frequency bins and all spectral slices are checked until the maximum number of records on SLINES.DAT is found.

INPUT

NOP - (in COMMON /CHUNKS/) max. number of peaks per power spectrum

DPO:TRKABL.DAT - (FORTRAN unformatted direct access file)
The file containing peaks (frequency bin numbers) determined in each of the selected power spectra. There are 1024 records of length NOP+2.

Typical record: LDUMY - value not used in XSTRPS
LINUM - index of spectral slice in MORPK2.DAT
PEAKPS - vector of bin numbers (in increasing order) of peaks in the associated power spectrum. NOP maximum, with zero fill.

Last record: (5 values)
ZSTART
ZMOVE cf SELECT File Descriptions
NWINDS
PTSDPY
TOT99 - number of spectral slices on file

SUBROUTINE NAME XSTR8S (Cont's)

OUTPUT

DPO:SLINES.DAT - (FORTRAN unformatted direct access file; 1024 records of size=3 words)

Typical record - LSTR8 (1) - frequency bin number
LSTR8 (2) - spectral slice index of start
LSTR8 (3) - number of consecutive slices with peak in this bin

Record #1024 - actual number of typical records

SUBROUTINES CALLED None

CALLED BY SELECT

CALLING SEQUENCE CALL XSTR8S

SUBROUTINE NAME XSTAR

PURPOSE To find a region in which to begin tracking where peaks are stable over several spectral slices.

DESCRIPTION XSTAR examines a specified triple in the file, DPO:SLINES.DAT, generated by XSTR8S in order to find an area where peak data are either unchanging or change very little over several consecutive spectral slices. Such a region is the start point for frequency tracking performed by subroutine XSETF3.

If the length of the constant frequency exceeds an input threshold (KNTLIM), the area described by the triple is called a vertical stable region, and the frequency bin number and index of the middle spectral slice are returned by XSTAR as the start of a track. If the length of the constant frequency does not exceed the stability threshold, XTSTAR tests for a diagonal region about the frequency and spectral slice obtained from SLINES.DAT. XTSTAR first searches succeeding spectral slices for an occurrence of a frequency bin peak no more than one bin away from the peak in the preceding slice. XTSTAR will do this search as far as it can, return to the start point, and perform a similar search backwards over spectral slices. The region is bounded by spectral slices whose closest peaks to the last frequency bin of the run differ by more than one. If the length of this "diagonal" region exceeds KNTLIM, the frequency bin and slice number of the midpoint are returned by XTSAR as a track start point.

If neither vertical region or diagonal regions are found, a start failure indicator is returned.

INPUT

Arguments

KROW - XSTR8S triple index

TOT99 - number of spectral slices on the file TRKABL.DAT

KNTLIM - minimum length of stable start threshold

DPO:SLINES.DAT cf. XSTR8S output

SUBROUTINE NAME XSTSAR (Cont'd)

OUTPUT

Arguments

BFREQ - frequency bin number of start of track

TIMBEG - index of start spectral slice

LGENUF - track start indicator

=1 - vertical region

=2 - diagonal region

=0 - no start

SUBROUTINES CALLED XNEARP

CALLED BY XSETF3

CALLING SEQUENCE CALL XSTAR (KROW,TOT99,KNTLIM,BFREQ, (BEG,LGENUF)

SURBROUTINE NAME XSETF3

PURPOSE To perform tracking of frequency peaks throughout the selected spectral slices.

DESCRIPTION XSETF3 is the master tracking routine which makes all necessary file assignments, accesses tracking parameters, performs dataI/O, and sequences routines to track frequency peaks throughout the ensemble of spectral slices selected as input by program SELECT. XSETF3 calls subroutine XTSTAR to find a stable region where it begins tracking at the middle spectral slice.

Tracking per se is accomplished by incrementing spectral slice indices by one, first going in a forward (+1) direction, then by returning to the start point and proceeding backwards (increment=-1). XSETF3 calls subroutine XNEARP to find the frequency bin in the next spectral slice which is closest to the current track frequency. If the change in bin numbers returned by XNEARP is within the frequency jump threshold (FROGAP), the new bin number is added to the track. If the FROGAP threshold is exceeded, XSETF3 looks ahead at successive spectral slices to see if the current frequency picks up again. If the peak resumes (within FROGAP) within the next TIMGAP slices, the track is continued and the gap is filled by a straight line interpolation between the existing bin numbers as provided by subroutine XINTER. Failure of the peak to resume within this TIMGAP interval signals the end of the current track.

Track data are stored in a 512-word vector, THSTRK, such that the content of THSTRK(I) is the frequency bin number for the spectral slice index, I. Spectral slices in which the track doesn't exist are designated by -1's in the corresponding THSTRK elements. The entire THSTRK vector for each separate track is stored on the disk file DPO:TRAKED.DAT, which has been set up to hold a maximum of 100 tracks. Before being stored, each track is compared to all other stored tracks by subroutine XMATCH. Duplicate tracks are not stored. The fact that duplicates are not determined until the end of a track, and the fact that two different tracks which intersect each contain the entire duplicated segment, represent significant shortcomings in this tracking scheme.

A flowchart for XSETF3 is contained in Figure 20.

SUBROUTINE NAME XSETF3 (Cont'd)

INPUT

COMMON /CHUNKS/SEGSIZ,NOP

SEGSIZ - not used

NOP - maximum number of peaks per spectral slice

DPO:TPARAM.DAT - (unit98) (unformatted binary) A single four-word record containing the current values for tracking parameters and thresholds

TRACKS - number of tracks requested

FROGAP - frequency jump threshold

KNTLIM - length threshold for stable region in which to start track

TIMGAP - spectral slice gap threshold for interpolation

DPO:SLINES.DAT - (unit88) (unformatted FORTRAN direct access) Contains line tracks of constant frequency; cf XSTR8S output

DPO:TRKABL.DAT - (unit 99)(unformatted FORTRAN direct access) Contains lists of peaks for each spectral slice; cf. XSTR8S input

OUTPUT

DPO:TRAKED.DAT - (unit11) (unformatted FORTRAN direct access) Contains all separate tracks through ensemble of spectral peaks up to TRACKS, the maximum number requested (100 max.). There are 101 records of 512 words each.

Typical record

(THSTRK (i),i=1,512) - list of frequency bin numbers or -1 if no track element exists in the spectral slice of corresponding index

Record #101: (6 words)

TRKTOT - number of separate tracks on file

TOT99 - number of spectral slices

PTSDPY

ZSTART cf. SELECT File Descriptions

ZMOVE

NWINDS

SUBROUTINE NAME XSETF3 (Cont'd)

SUBROUTINES CALLED XTSTAR

XNEARP

XINTER

XMATCH

CALLED BY SELECT

CALLING SEQUENCE CALL XSETF3

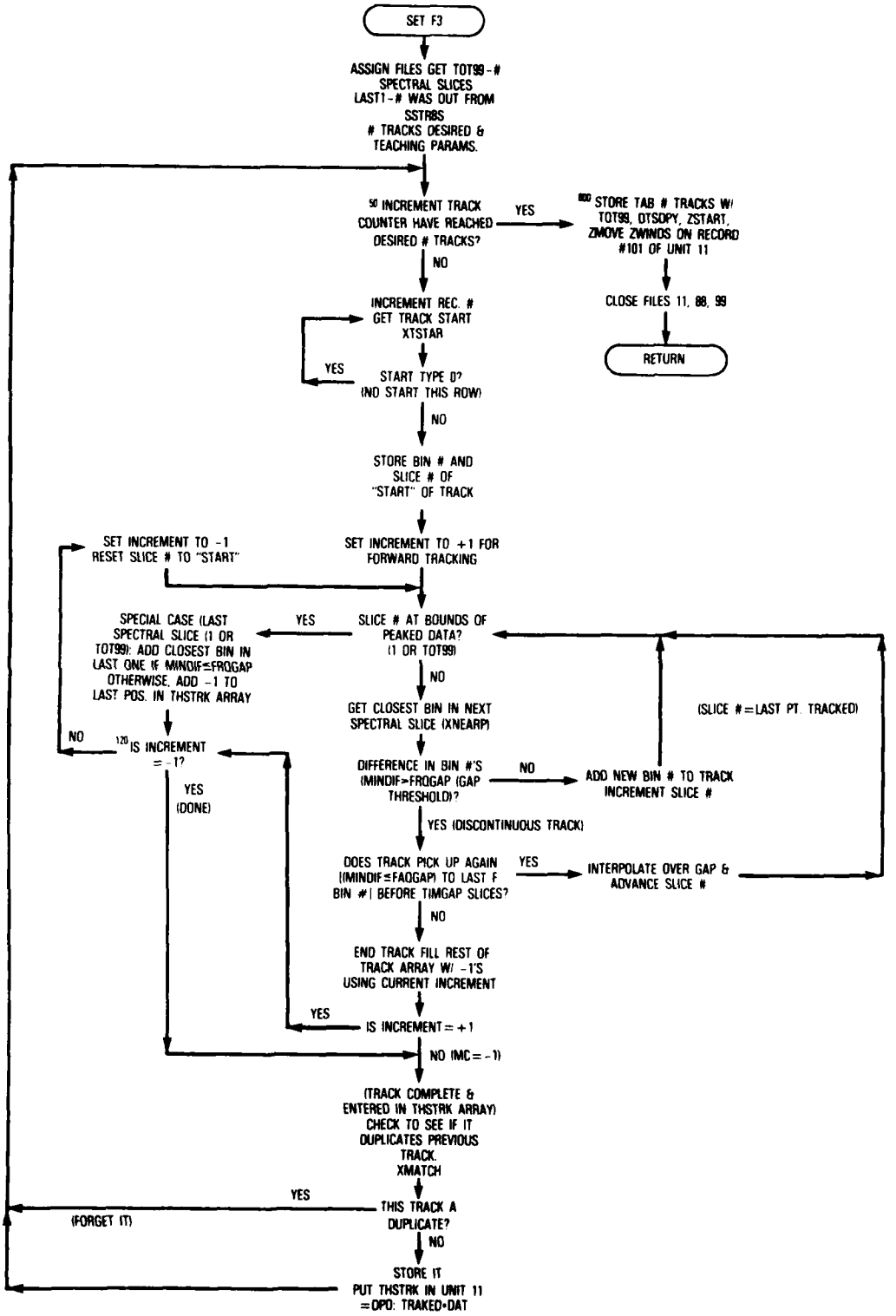


Figure 20 - Subroutine SETF3 Flowchart

SUBROUTINE NAME XNEARP

PURPOSE To locate the peak closest to an input frequency bin number and compute the absolute difference in bin numbers for a given spectral slice.

DESCRIPTION XNEARP is called with two input arguments containing the index of the desired spectral slice and the number of the basis frequency. XNEARP reads the peak bin numbers for the spectral slice from the direct access file DPO:TRKABL.DAT. If there are no peaks in the spectral slice, bin number zero is output as the closest peak with absolute difference equal to 32000. If there are peaks, the closest bin number is selected and the absolute difference computed.

INPUT

Arguments

FREQ - input (base) frequency bin number

TSLICE - index of spectral slice data to be examined

DPO:TRKABL.DAT - cf. XSTR8S, Input

OUTPUT

Arguments

NFREQ - bin number of closest peak in given spectral slice

MINDIF - absolute difference in frequency bin numbers, /FREQ-NFREQ/

SUBROUTINES CALLED None

CALLED BY XTSTAR, XSETF3

CALLING SEQUENCE CALL XNEARP (FREQ,TSLICE,NFREQ,MINDIF)

SUBROUTINE NAME XINTER

PURPOSE To interpolate frequency bin numbers over a gap in a frequency track.

DESCRIPTION Subroutine XINTER has as input a partial track of frequency peaks which has a gap or interval of spectral slice indices without any peaks belonging to the track. The indices bounding this interval are input to XINTER along with the frequency bin numbers in the track at those indices. XINTER determines the change in bin numbers divided by the difference in indices and, using this value as the slope of a straight line between two points, computes the frequency bin determined by that line at each intervening index and adds it to the track. Bin numbers are computed in floating point arithmetic and rounded to the nearest integer.

INPUT

Arguments

THSTRK - a 512-word vector containing the partial track

FREQ - old bin number

TIMPTR - old spectral slice index

NFREQ - new bin number

TPR - new spectral slice index

OUTPUT

Arguments

THSTRK - a 512-word vector containing augmented frequency track

SUBROUTINES CALLED None

CALLED BY XSETF3

CALLING SEQUENCE* CALL XINTER (THSTRK,KTRAK,INC,FREQ,TIMPTR,NFREQ,TPR)

*Note: Arguments KTRAK and INC are not used.

SUBROUTINE NAME XMATCH

PURPOSE To compare the input track to all tracks already obtained to check for duplicate tracks.

DESCRIPTION The current track, as input to XMATCH, is contained in a 512-word vector and all tracks previously obtained are stored on disk. XMATCH reads in each track from the disk file into another vector, local to XMATCH, and simply compares the two vectors word-by-word. If the current track matches no stored track, the total number of tracks is incremented and a no-duplicate indication is returned. If the current track does match a previous track, the current track index is decremented and a duplicate indication is returned to the calling program.

INPUT

Arguments

THSTRK - vector containing current track

TOT99 - maximum number of spectral slices in input data

KTRAK - index id of current track

TRKTOT - current total number of different tracks on file

DPO:TRAKED.DAT - cf XSETF3, output

OUTPUT

Arguments

TRKTOT - new total of different tracks on file

DUPE - LOGICAL * 1 variable which is true only when input track duplicates a previously stored track

SUBROUTINES CALLED None

CALLED BY XSETF3

CALLING SEQUENCE CALL XMATCH (THSTRK,TOT99,KTRAK,TRKTOT,DUPE)

SELECT FILE DESCRIPTIONS

DPO:MORPK2.DAT

DESCRIPTION MORPK2.DAT is the spectral data input file to program SELECT generated by program VU. It contains a maximum of 1023 spectral slices, each of which consists of a list of frequency bin numbers of peaks in an associated power spectrum. An additional record on the file contains specific parameter information concerning the power spectra taken in VU.

TYPE MORPK2.DAT is a FORTRAN unformatted direct access file.

CONTENTS

Records - 1023: LTYPE, (PPDAT(I),I = 1,10)

- LTYPE - integer equal to 1 or 2 to specify type of baseline (solid or broken) used to display spectral slice
- PPDAT - list of integer frequency bin numbers

Records - 1024: ZSTART, ZMOVE, NWINDS, PTSDPY

- ZSTART - floating point value of start point in raw data file of first power spectrum
- ZMOVE - floating point value of the jump in number of points between power spectra
- NWINDS - integer value of the total number of spectral slices on this file
- PTSDPY - number of points taken into consideration for each power spectrum

DPO:TPARM.DAT

DESCRIPTION TPARAM.DAT is an input/output file which contains the working values and default values of the seven user modifiable parameters to program SELECT.

TYPE TPARAM.DAT is an unformatted binary FORTRAN file.

CONTENTS

Words 1-7 are integer values of the working parameters.

- TRACKS - number of tracks requested
- FROGAP - frequency jump threshold
- KNTLIM - track length start threshold
- TIMGAP - spectral slice gap threshold
- CMAND1 - automatic move direction (up, down)
- NUM1 - amount of automatic move
- ISTART - lower frequency bound for displays

Words 8-14 contain the integer default values of the same parameters.

DPO:TRKABL.DAT

DESCRIPTION TAKABL.DAT is the file containing the spectral slice data from MORPK2.DAT which have been selected for analysis by the tracking procedures.

TYPE TRKABL.DAT is a FORTRAN unformatted direct access file.

CONTENTS

Records 1-1023: LTYPE, LINUM (PPDAT(I), I=1, 10)

- LTYPE - an integer 1 or 2 to specifying whether the baseline used to display the spectral slice is solid or broken line
- LINUM - integer spectral slice index in MORPK2.DAT
- PPDAT - list of integer frequency bin numbers

Record 1024: ZSTART, SMOVE, NWINDS, PTSDPY, TOT99

- The first four values are same as for MORPK2.DAT
- TOT99 - number of spectral slices in TRKABL.DAT

DPO:TRAKED.DAT

DESCRIPTION TRAKED.DAT is the file containing all tracks generated by the tracking procedures operating on spectral data contained in DPO:TRKABL.DAT. The number of tracks is limited to the smaller of a user specified limit, TRACKS, contained in TPARAM.DAT and 100, the programmed size limit for TRAKED.DAT.

TYPE TRAKED.DAT is a FORTRAN unformatted direct access file.

CONTENTS

Records 1-100: (THSTRK(I), I = 1, 512)

THSTRK - list of bin numbers representing peaks in the track or -1 where no track element exists in the spectral slice of corresponding index

Record 101: TRKTOT, TOT99, PTSDPY, ZSTART, ZMOVE, NWINDS

(cf. TRKABL.DAT and MORPK2.DAT)

TRKTOT - actual number of tracks on file

DPO:SLINES.DAT

DESCRIPTION SLINES.DAT is a scratch file used by the tracking procedure in the track start testing. SLINES.DAT stores all areas of straight line tracks, i.e., identifies regions in which a constant peak exists.

TYPE SLINES.DAT is a FORTRAN unformatted direct access file.

CONTENTS

Records 1-1023: (LSTR8(I), I = 1, 3)

LSTR8(1) = frequency bin number

LSTR8(2) = spectral slice index of start

LSTR8(3) = number of consecutive slices with a peak in this bin

Record 1024 - actual number of LSTR8

SELECT LINKING PROCEDURE

The necessary command to link program SELECT on the PDP 11/45 is shown in Figure 21.

```
.RUN LINK
*DPO:SELECT,SELECT=DPO:SELECT,SY:VTLIB,LPSLIB,SYSLIB/F/C
DPO:MARKIT/O:1/C
DPO:MOVE2/O:1/C
DPO:XPEAKS/O:1/C
DPO:XSTR8S/O:1/C
DPO:XSETF3/O:1/C
DPO:XSTAR,XNEARP/O:2/C
DPO:XINTER/O:2/C
DPO:XPLOT,XYHDLR/O:2/C
DPO:BUNCH/O:2/C
DPO:XBLANK/O:2/C
DPO:POTTY/O:2/C
DPO:XTDISP/O:2/C
DPO:WHERE/O:2/C
DPO:XMATCH/O:2/C
DPO:XUP/O:2/C
DPO:XADJST/O:2
*
```

Figure 21 - Program SELECT Linking Procedure

PROGRAM WAVAN

WAVAN is a syntactic pattern learning/classification system for processing one-dimensional signal waveforms. These waveforms are strings of symbols drawn from a given alphabet (e.g., RISE, FALL, LEVEL), each with associated parameters (like intensity, frequency, duration, decay time) derived from such routines as found in program VU discussed in the first section of this report. The functional overview of WAVAN is contained in the ISPARS User's Guide.[1] This section details the program description, input/output, files, and linking procedure for WAVAN.

WAVAN SUBROUTINE DESCRIPTIONS

SUBROUTINE NAME WAVAN

PURPOSE To initiate waveform analysis learning or classification subsystem as specified by the user.

DESCRIPTION WAVAN is a GIRL program[5] which first initializes variables and arrays associated with the wave feature transition graph data structure and tests keyboard input to determine the type of waveform processing to be performed. The user first specifies whether processing is to be in the test or train mode. In the test mode, WAVAN immediately reads from disk the stored feature graph and the current states of the associated variables. The waveform data to be classified are then read and subroutine TEST is called to perform the classification.

INPUT

Keyboard

YES or NO to test question

YES or NO to create question if previous answer NO

Disk

DPO:WVTREE.NEW - File containing waveform feature transition graph; if not, create run

DPO:SIGDAT.NEW - File containing test or training signal waveforms

OUTPUT

Test signal classification if in test mode

Disk:

DPO:WVTREE.NEW - Updated waveform feature transition graph if in train mode

CALLED BY (This is a main program - not called by any other.)

SUBROUTINES CALLED

GROW

TEST

LVFECH
LVDUMP GIRS I/O routines[4]

SUBROUTINE NAME GROW

PURPOSE To create/augment the waveform feature transition graph and library of average waves for a given event.

DESCRIPTION Subroutine GROW calls WTWAV and DESCEN to assign feature weights and reorder the feature string according to confidence (weight). Features from the recorded sequence are taken one by one to form partial strings, each with a cumulative confidence. When the confidence of a partial string exceeds a threshold, the last feature added is added as a link in the feature transition graph and the event represented by this feature is added to a list tied to the graph at this point. If the event is already on the list, its associated confidence (also on the list) is increased and the temporal position average for the feature is updated. This average associates a relative time of occurrence with each feature. When the sample feature string for an event is exhausted, GROWAV is called to store the wave data for the event in the waveform library on disk.

INPUT

Arguments

EVNTID - id number of sample event in event dictionary

Common /FEATUR/

MLAS - array of triples, each representing a waveform feature

NFORMS - length of MLAS

/PRMTRS/ - list of parameters to be used in GROW and GROWAV

/WORDS/ - list of random numbers associated with dictionary events for use in GIRL graph

/TRE/ - list of random numbers associated with content and structure of GIRL graph structure used for storage of feature transition graph

OUTPUT Modified feature transition graph

CALLED BY WAVAN

SUBROUTINES CALLED WTWAV, DESCEN, GROWAV, STRING

CALLING SEQUENCE CALL GROW (EVNTID)

SUBROUTINE NAME GROWWAV

PURPOSE To generate a list of different average waves representing a given event.

DESCRIPTION GROWAV obtains the node containing the list of representative waves for a given event and calls COMPWV to compare a new (input) wave with each stored wave on the list. COMPWV returns the identity of the closest match and, if the score of the match is acceptable, GROWAV averages the new wave with the specified matching wave on the list and replaces the stored wave with the new average. If the score of match is not acceptable, the new wave is added to the list as a new representative wave for the given event.

INPUT

Arguments

EVNTID - event id

Common /WORDS/

WORDS - array of random numbers forming nodes in GIRL graph data base representing each event

Common /WAVFRM/

WAVPTR - next available block on disk file DPO:WAVDIR.NEW for storing waves

OUPUT Updated or averaged waveforms stored on GIRL graph data base

CALLED BY GROW

SUBROUTINES CALLED

COMPWV

OUTWAV

GETWAV

CALLING SEQUENCE CALL GROWAV (EVNTID)

SUBROUTINE NAME TEST

PURPOSE To classify a test waveform by generating an ordered list of possible identifications.

DESCRIPTION The test waveform string is recorded and processed in order of individual wave feature reliabilities. The recorded string of features is compared with a stored syntax of wave features characteristic of the library of stored waveforms. Cumulative confidences are maintained as the syntax is traversed, and when confidence thresholds are exceeded, a list of possible matching library elements (called candidates) is formed. If the principal confidence thresholds are not exceeded, a backup (temporary) list of candidates exceeding a lower threshold is also maintained. If the candidate list is too large (> WTHRES), subroutine TESTWV is called to compare waveforms more closely and reorder the list; the top WTHRES candidates are returned as output. If the candidate list is too short (< WTHRES), the temporary list is used to augment the candidate list. If no valid candidates are found, subroutine WAVES is called to compare the test wave with the entire library. Figure 22 contains a flowchart for subroutine TEST.

INPUT

Argument

MLAS - test waveform array

NFORMS - length of MLAS

DISK

DPO:WVTREE.NEW - contains the GIRL graph storing the waveform syntax and also the variables associated with the graph

OUTPUT

Common /MATR/

MATCH - vector of id numbers of wave matches

MATES - number of matches

CONMAT - vector of confidences of associated matches

CALLED BY This is called by main program.

SUBROUTINE NAME TEST (Cont'd)

SUBROUTINES CALLED

WTWAV

DESCEN

MAXSTK

TESTWV

WAVES

SETLIB

STRING

GIRS routines

CALLING SEQUENCE CALL TEST (MLAS, NFORMS)

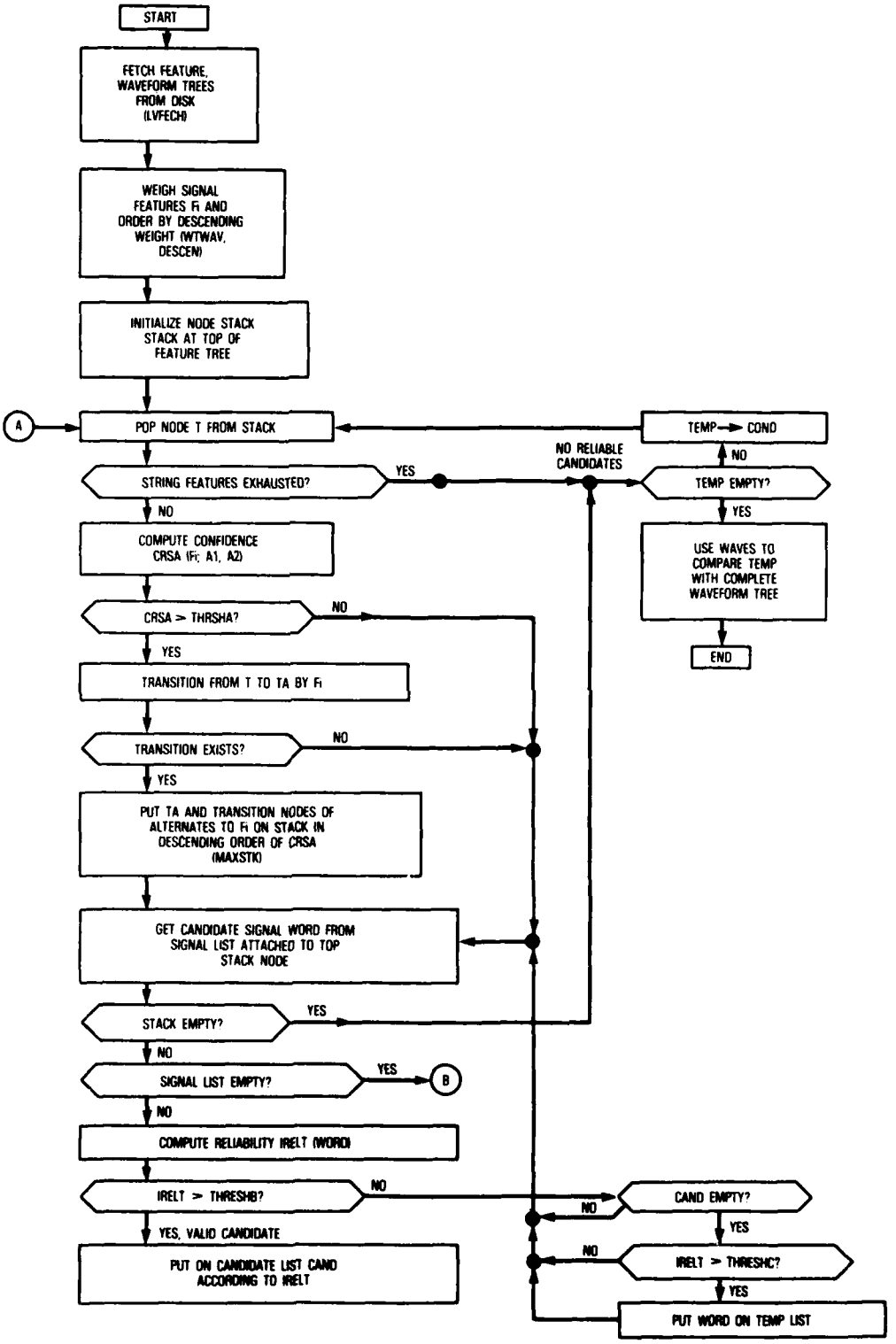


Figure 22 - Subroutine TEST Flowchart

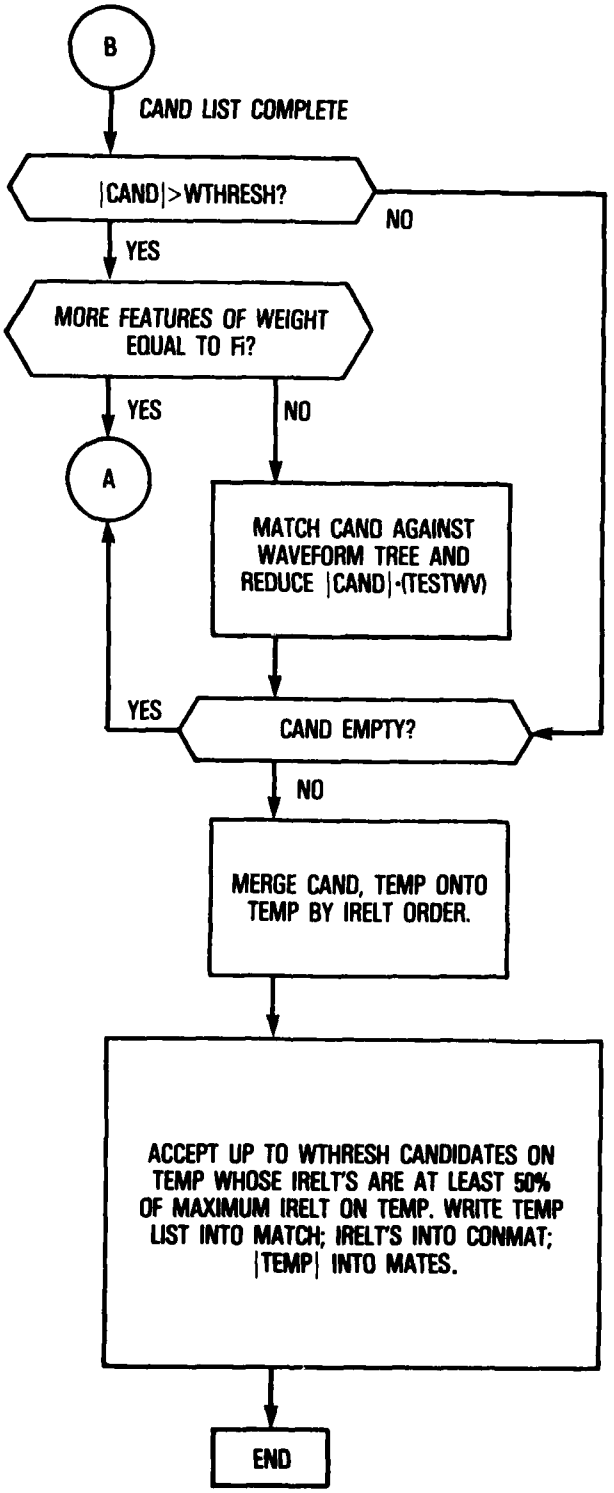


Figure 22 - Subroutine TEST Flowchart (Cont'd)

SUBROUTINE NAME TESTWV

PURPOSE To compare waveforms of all elements on candidate list to test form and save list matches on a final candidate list.

DESCRIPTION A list of candidate matches is supplied to TESTWV, and TESTWV calls COMPWV to compare candidate forms to the test form and score the match. The candidates are recorded on a temporary list in order of scores and the best matches (number equal to an input parameter, WTHRES) are re-created on a new candidate list.

INPUT

Arguments

MLAS - test waveform string

NFORMS - length of MLAS

Other

Candidate list contained in the GIRL graph linked to the node CAND by LIST and RELIST. CAND, LIST, RELIST contained in COMMON/TRE/.

OUTPUT Revised candidate list contained in the GIRL graph

CALLED BY TEST

SUBROUTINES CALLED COMPWV

GIRS routines

CALLING SEQUENCE CALL TESTWV (MLAS, NFORMS)

SUBROUTINE NAME WAVES

PURPOSE To cycle through entire library of stored waveforms to compare each with a given test form.

DESCRIPTION WAVES cycles through the vocabulary of stored waveforms, obtains the head node of associated waveforms, and calls subroutine COMPWV to compare stored forms with test form input to WAVES. Stored forms whose score is below a program threshold are retained on a candidate list of matching forms.

INPUT

Arguments

MLAS - waveform string

NFORMS - length of MLAS

OUTPUT Candidate list in GIRL graph of matching forms and score for match

CALLED BY TEST

SUBROUTINES CALLED COMPWV

GIRS routines

CALLING SEQUENCE CALL WAVES (MLAS, NFORMS)

SUBROUTINE NAME MAXSTK

PURPOSE To put a node on the stack such that the stack lists items in order of decreasing confidence.

DESCRIPTION The list attached to STACK by the RELIST link is scanned until an element is found whose confidence is below that of the input item; the new item and its confidence are added to the list at this point.

INPUT

Arguments

TA - form to be added to stack

CRSA - confidence associated with this node

I - position of the form in original waveform

GIRL waveform graph

OUTPUT Updated GIRL waveform graph

CALLED BY TEST

SUBROUTINES CALLED GIRS system routines

CALLING SEQUENCE CALL MAXSTK (TA, CRSA,I)

SUBROUTINE NAME COMPWV

PURPOSE To compare an input wave to each wave associated with an input node.

DESCRIPTION COMPWV first condenses the input wave by eliminating levels as separate forms and adding the duration of levels to the duration of the preceding form. The same thing is done for forms representing small changes. The condensed wave is compared to each stored wave form by form. If forms do not agree, COMPWV attempts to either stretch or compress the input form to achieve agreement. Forms agreement is measured by the in-line function, DELTST, which is a weighted average of the difference in level, decay time, and duration. These three differences are also summed over all forms for use in a scoring function to determine the measure of match between the input and stored waves. This score is also an in-line function, SCORIT, computing a weighted average of the difference sums. Stretching is achieved by repeating a waveform, and compression is achieved by deleting a waveform and either case is accompanied by appropriate modification to the differences and difference sums.

INPUT

Arguments

NODE - random number associated with node in wavegraph to which library (stored) waves are linked

PREV - current best score in waveform comparisons

MLAS - input wave form vector

NFORMS - number of form in input wave

TSTING - true for test mode, false for training

Other

GIRL graph storing waveform data base

OUTPUT

Arguments

PREV - score for this wave comparison if stored wave resulted in better score than previous best

CHOI - index of form on input node list that corresponds to output score

Common /AVWAV/

AVLEN - length of condensed form

AVWAV - new wave generated by compressing/stretching

SUBROUTINE NAME COMPWV (Cont'd)

CALLED BY

WAVES

TESTWV

GROWAV

SUBROUTINES CALLED GETWAV

GIRS routines

CALLING SEQUENCE CALL COMPWV (NODE, PREV, CHOI, TSTING)

SUBROUTINE NAME DESCEN

PURPOSE To determine that permutation of subscripts which reorders elements of the feature string in terms of decreasing confidence (weight).

DESCRIPTION The weight vector associating weights or reliability measures with the string of feature elements is scanned repeatedly for highest weight; each time the index is added to the permutation matrix REORD.

INPUT

Common /FEATUR/KLAS (768,3), NFEAT
/WEIGHT/WT(768)

where

KLAS is feature string of up to 768 features

NFEAT is number of features in string

WT is vector of associated feature weights

OUTPUT REORD - vector of permutation of weight indices (argument to DESCEN)

CALLED BY

GROW

TEST

SUBROUTINES CALLED None

CALLING SEQUENCE CALL DESCEN (REORD)

SUBROUTINE NAME WTWAV

PURPOSE To assign weights to feature string (waveform) elements.

DESCRIPTION Zero weights are assigned to the first and last form (features) in the string. The weights for rises or falls are

$$2048\ (D)/M$$

where D is the absolute change in level for this form and M is the maximum of D for all rises and falls in the string. The weight for a level is

$$2048\ (L)/G$$

where L is the duration of the level and G is the maximum G for all levels.

INPUT

Arguments

MLAS - waveform string

NFORMS - length of MLAS

OUTPUT

Common

WT - vector of weights associated with MLAS

CALLED BY

GROW

TEST

SUBROUTINES CALLED None

CALLING SEQUENCE CALL WTWAV (MLAS, NFORMS)

INTEGER FUNCTION STRING

PURPOSE To determine the GIRL graph link to be associated with a given form (feature).

DESCRIPTION The value of the function STRING is set to the GIRL link random numbers RISE, FALL, or LEVEL to correspond to the type of form referred to by the input argument, K.

INPUT

Arguments

K - form pointer

MLAS - vector of waveforms

Common /TRE/

RISE, FALL, LEVEL - random numbers identifying links associated with waveform types

OUTPUT STRING (K, MLAS) = either RISE, FALL, or LEVEL depending on the type of the k^{th} waveform

CALLED BY

GROW

TEST

SUBROUTINES CALLED None

CALLING SEQUENCE CALL STRING (K, MLAS)

SUBROUTINE NAME GETWAV

PURPOSE To read a waveform from disk library into program array.

DESCRIPTION GETWAV reads PDP-11 System Library of binary data from disk using channel and block index supplied to it through arguments.

INPUT

Arguents

INDEX - block number index of desired waveform on disk

ICH - channel number assigned to waveform input file

DISK

DPO:WAVDIR.NEW - binary file containing directory of stored waveforms. First two words are number of forms and total decay.

OUTPUT

Common /WAVFRM/

SLNGTH - length of stored waveforms

DECAY - total decay of waveform

STDWAV(I,3) - stored waveform

CALLED BY CPMPWV

SUBROUTINES CALLED IREADW of PDP-11 SYSLIB

CALLING SEQUENCE CALL GETWAV (INDEX, ICH)

SUBROUTINE NAME OUTWAV

PURPOSE To store waveforms on disk.

DESCRIPTION OUTWAV performs a PDP-11 System Library write to disk using channel number and block index supplied as arguments.

INPUT

Arguments

INDEX - block number of desired waveform on disk

ICH - channel number assigned to waveform directory file

Common /WAVFRM/

SLNGTH - length of waveform array

DECAY - total decay for waveform

STDWAV - stored waveform

OUTPUT

DISK

DPO:WAVDIR.NEW - binary file containing directory of stored waveforms

CALLED BY GROWAV

SUBROUTINES CALLED None

CALLING SEQUENCE CALL OUTWAV (INDEX, ICH)

SUBROUTINE NAME SETLIB

PURPOSE To set up file definitions for use within rest of program.

DESCRIPTION PDP-11 System Library functions are used to establish the file DPO:WAVDIR.NEW - create it if this is a "create" or "test" run or find it if this is an "update" run. In addition, logical unit number 99 is assigned to DPO:WVTREE.NEW.

INPUT

Arguments:

CREATE - logical variable set to .TRUE. for create run, .FALSE. for update run

NBLKS - number of blocks to request for each record or waveform on disk

OUTPUT None

CALLED BY WAVAN

SUBROUTINES CALLED PDP-11 SYSLIB

CALLING SEQUENCE CALL SETLIB (CREATE, NBLKS)

WAVAN FILE DESCRIPTIONS

DPO:WVTREE.NEW

DESCRIPTION WVTREE.NEW is the file containing the waveform feature transition graph and related variables. It is strictly an output file when WAVAN is executed in the "create" training mode. It is both an input and an output file in the training update mode, and only an input file in the test mode.

TYPE WVTREE is a FORTRAN unformatted binary file

CONTENTS The contents of WVTREE are read by a call to LVFECH[4].

The last logical record on the file contains program parameters related to the graph data and is read as follows:

READ (99) FORMNT, WAVPTR, WAVE, INAV, TREE, LIST, NUM, DICTSZ, WORDS, RISE, FALL, LEV, PLACE, AELIST, TEMP, CAND, STACK

where

WORDS is dictionary of random numbers corresponding to events being classified by the graph. Current program limits number of events to twenty.

DPO:WAVDIR.NEW

DESCRIPTION WAVDIR.NEW is a file storing the directory of all stored waves. All significantly different waves encountered in the training process are stored in the directory.

TYPE WAVDIR.NEW is a PDP-11 System Library file of binary data currently limited to 600 blocks in length. Each waveform occupies 6 blocks or 1536 words.

CONTENTS

Each waveform record consists of

SLNGTH - number of triples specifying wave

DECAY - integer value typifying overall decay in level values over duration of wave

STWAV(512,3) - array of triples specifying triples wave store:

- o new height if form a rise or fall or zero if level
- o decay time if rise or fall or magnitude of level
- o duration of form

DPO:SIGDAT.NEW

DESCRIPTION SIGDAT.NEW is the file containing waveform data of known events that are to be used to train the waveform feature transition graph and build the waveform directory.

TYPE SIGDAT.NEW is a FORTRAN unformatted binary file.

CONTENTS The first logical record on SIGDAT.NEW consists of one integer value, TIMES, equal to the number of remaining logical records on file. The remaining logical records have the form

EVNTID, NFORMS (MLAS(I,J), J = 1, 3) I = 1, NFORMS)

where

EVNTID is the identification number of the event depicted by the wave

NFORMS is the number of triple waveforms constituting the wave

MLAS is the array of waveform triples in the wave

WAVAN LINKING PROCEDURE

Figure 23 contains the PDP 11/45 command to link WAVAN.

```
RUN LINK
DPO:VARY=DPO;VARY, OTHERS, SETLB2, GIRSZ, SYSLIB/F/C
DPO:ANAAAA/O:1/C
DPO:UPTTTT/O:1
```

Figure 23 - WAVAN Linking Procedure

REFERENCES

1. Parsons, W. et al, "Interactive Signal and Pattern Analysis and Recognition System (ISPARS). Users Guide," DTNSRDC Report (in process of being published).

2. Sanyal, P. K. et al, "The Waveform Processing System (WPS)," Rome Air Development Center Technical Report TR-76-224, Vol. I (Oct 1976), Vol. II (Sep 1976, Vol. III (July 1976), Vol. IV (Feb 1977).

3. Sammon, J. W., "The On-Line Pattern Analysis and Recognition System--OLPARS," Rome Air Development Center Technical Report TR-68-263 (1968).

4. Zaritsky, I., "GIRS (Graph Information Retrieval System) Users Manual," DTNSRDC Report 79/036 (April 1979).

5. Berkowitz, S., "Graph Information Retrieval Language; Programming Manual for FORTRAN Complement; Revision One," DTNSRDC Report 76-0085 (Feb 1976).

INITIAL DISTRIBUTION

Copies	
1	CNR
1	NRL
1	NUSC
12	DTIC

CENTER DISTRIBUTION

Copies	Code	Name
1	18	G. Gleissner
1	1805	E. Cuthill
2	1808	D. Wildy
1	182	A. Camara
5	1824	S. Berkowitz
1	1828	J. Garner
1	184	J. Schot
1	185	T. Corin
1	187	M. Zubkoff
1	189	G. Gray
1	522.1	TIC (C)
1	522.2	TIC (A)
1	93	L. Marsh

DTNSRDC ISSUES THREE TYPES OF REPORTS

1. DTNSRDC REPORTS, A FORMAL SERIES, CONTAIN INFORMATION OF PERMANENT TECHNICAL VALUE. THEY CARRY A CONSECUTIVE NUMERICAL IDENTIFICATION REGARDLESS OF THEIR CLASSIFICATION OR THE ORIGINATING DEPARTMENT.

2. DEPARTMENTAL REPORTS, A SEMIFORMAL SERIES, CONTAIN INFORMATION OF A PRELIMINARY, TEMPORARY, OR PROPRIETARY NATURE OR OF LIMITED INTEREST OR SIGNIFICANCE. THEY CARRY A DEPARTMENTAL ALPHANUMERICAL IDENTIFICATION.

3. TECHNICAL MEMORANDA, AN INFORMAL SERIES, CONTAIN TECHNICAL DOCUMENTATION OF LIMITED USE AND INTEREST. THEY ARE PRIMARILY WORKING PAPERS INTENDED FOR INTERNAL USE. THEY CARRY AN IDENTIFYING NUMBER WHICH INDICATES THEIR TYPE AND THE NUMERICAL CODE OF THE ORIGINATING DEPARTMENT. ANY DISTRIBUTION OUTSIDE DTNSRDC MUST BE APPROVED BY THE HEAD OF THE ORIGINATING DEPARTMENT ON A CASE-BY-CASE BASIS.


DAT
FILM